

## Integration of Multiple Services using Application Programming Interface

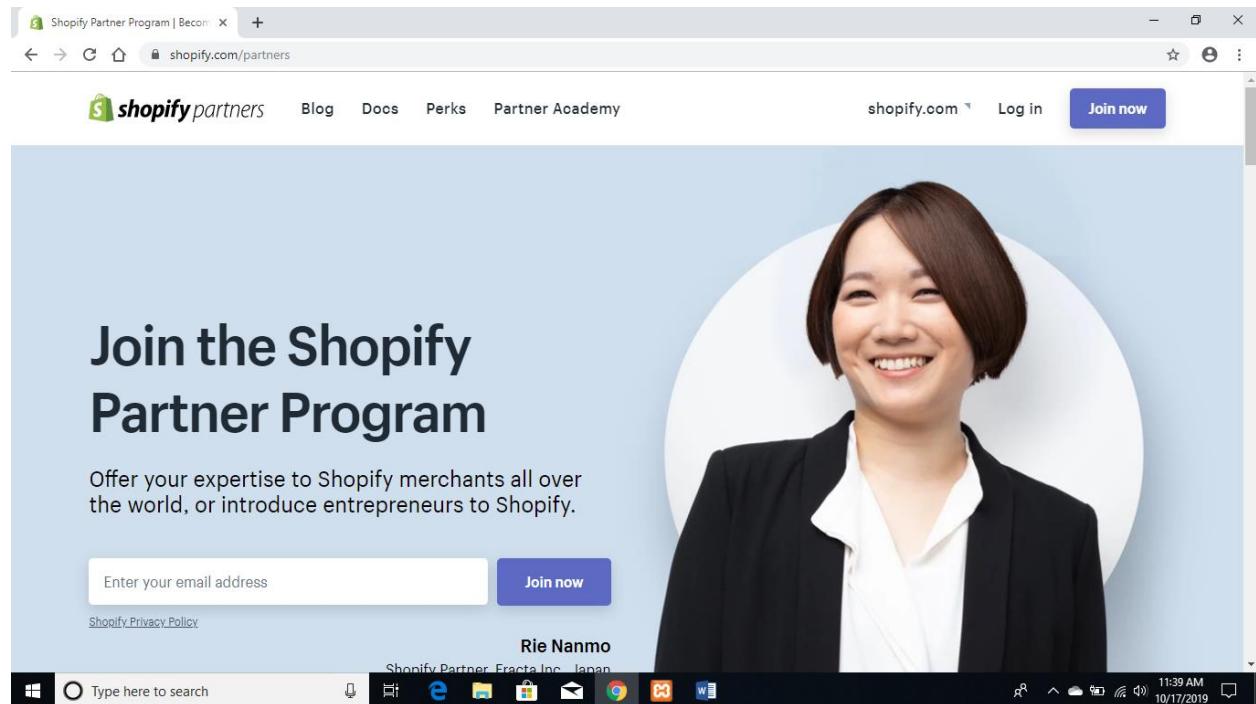
Platform: Shopify

Tools:

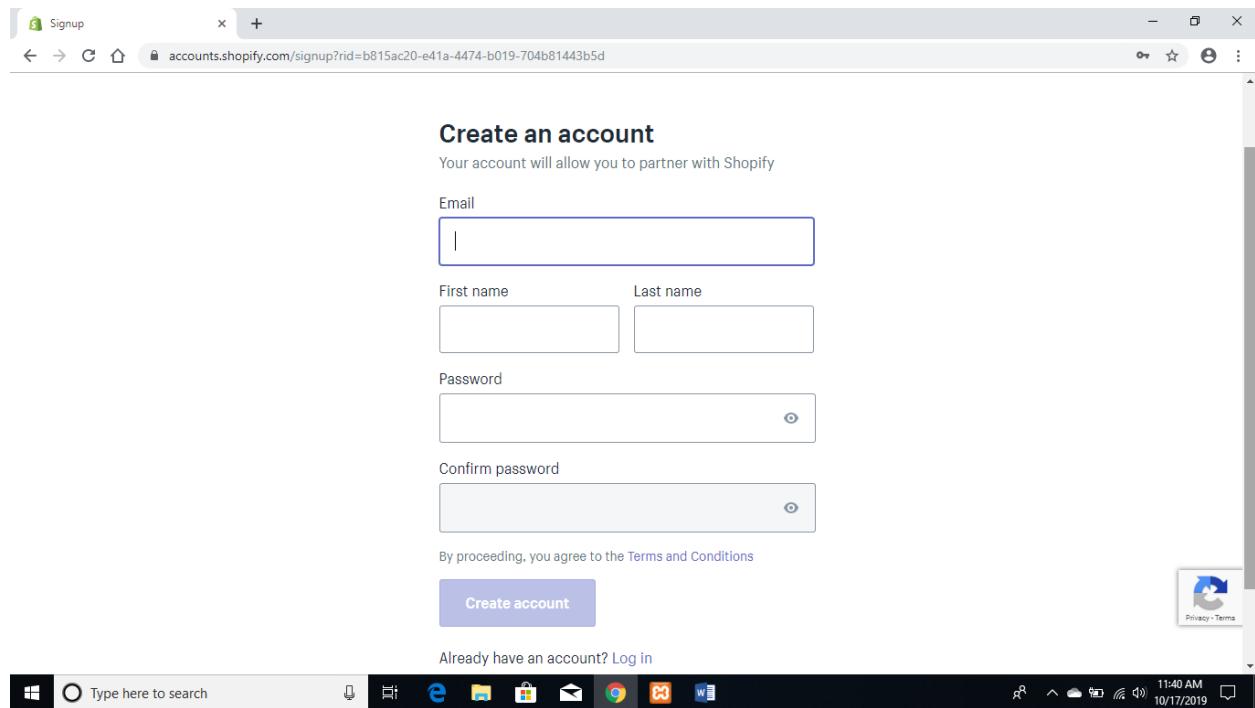
- a. ngrok
- b. nodejs (10.16.3 LTS)
- c. Postman

### **Task Number 1: Creating a Shopify Partner Account**

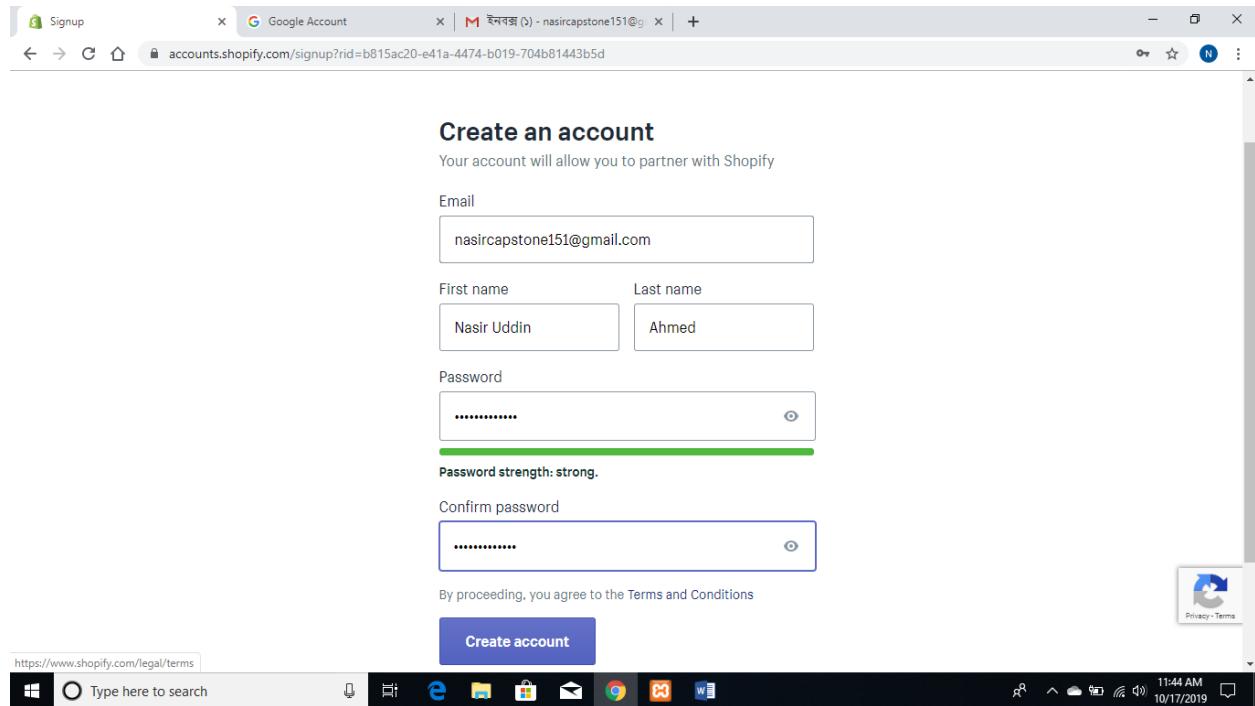
Go to <https://www.shopify.com/partners>



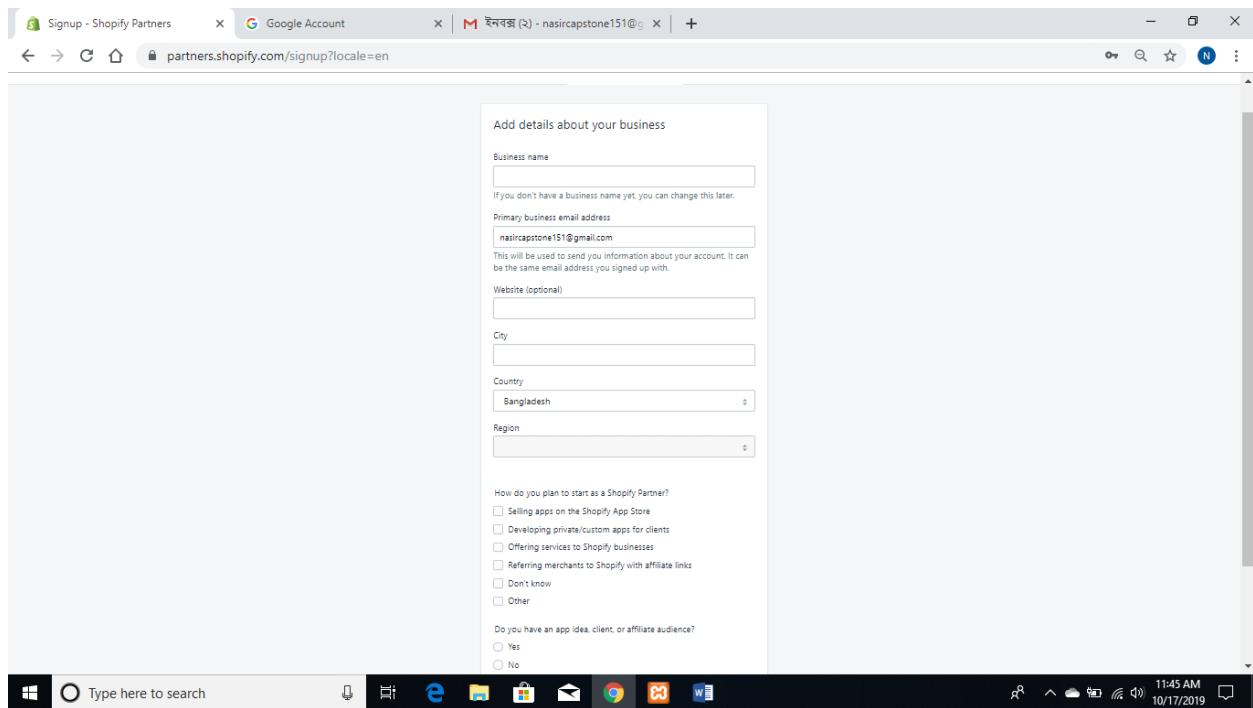
Press Join now



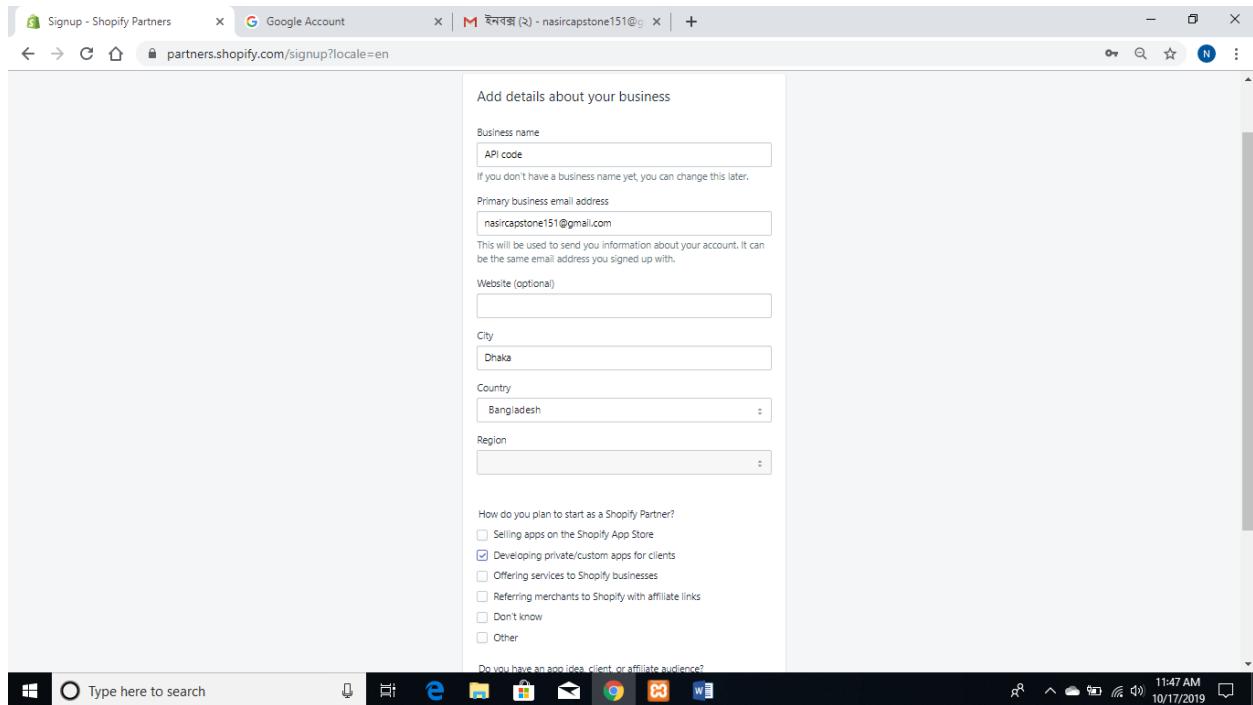
Feel up the input fields with necessary information. I have put my necessary information



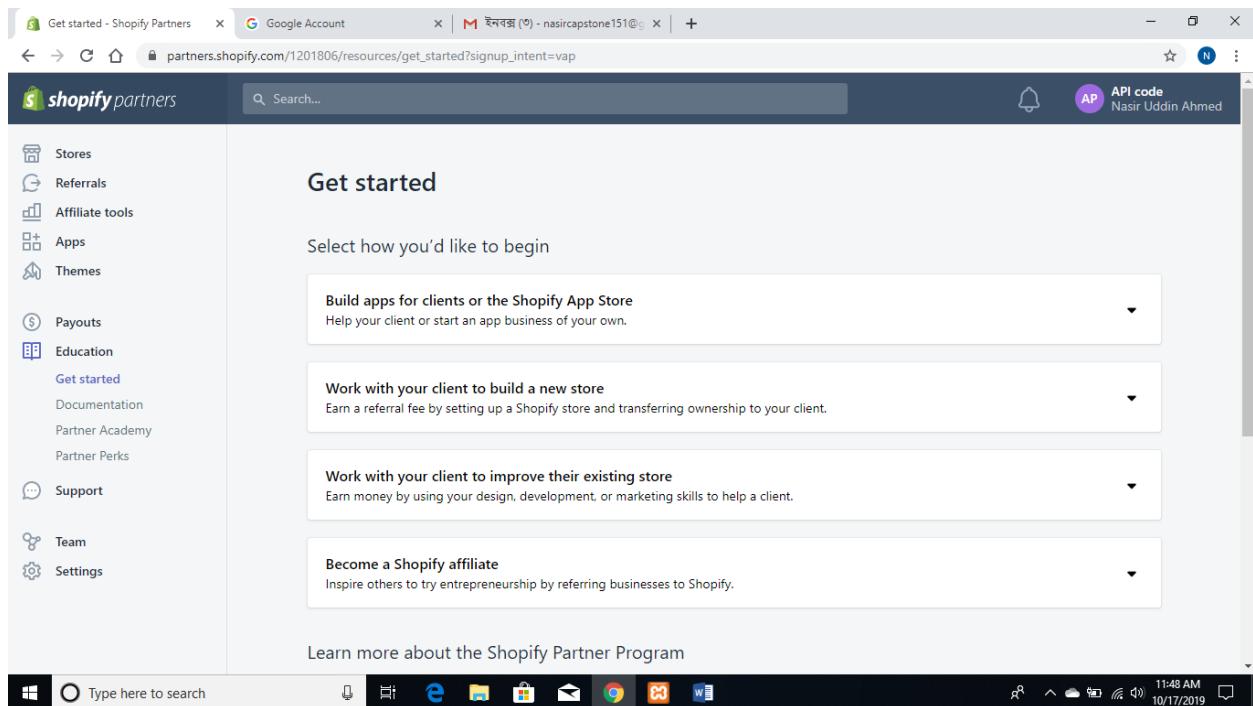
Press Create account



In this page put Business Name, City, How do you plan to start as a Shopify Partner?



Now press View your dashboard.



This is the Shopify(admin) dashboard.

Now we have a Shopify partner account, where we have a store. Now we will develop application.

### Task Number 2: Download nodejs, ngrok and postman

Go to <https://nodejs.org/en/download/>

The screenshot shows the Node.js Downloads page. At the top, there are tabs for HOME, ABOUT, DOWNLOADS, DOCS, GET INVOLVED, SECURITY, NEWS, and FOUNDATION. The FOUNDATION tab is highlighted. Below the tabs, it says "Downloads" and "Latest LTS Version: 10.16.3 (includes npm 6.9.0)". It encourages users to "Download the Node.js source code or a pre-built installer for your platform, and start developing today." There are two main sections: "LTS Recommended For Most Users" and "Current Latest Features". Under "LTS", there are links for "Windows Installer (.msi)", "Windows Binary (.zip)", "macOS Installer (.pkg)", and "macOS Binary (.tar.gz)". Under "Current", there are links for "32-bit" and "64-bit" versions of "node-v10.16.3.pkg" and "node-v10.16.3.tar.gz". The bottom of the page shows a Windows taskbar with the date and time as 10/17/2019 at 11:52 AM.

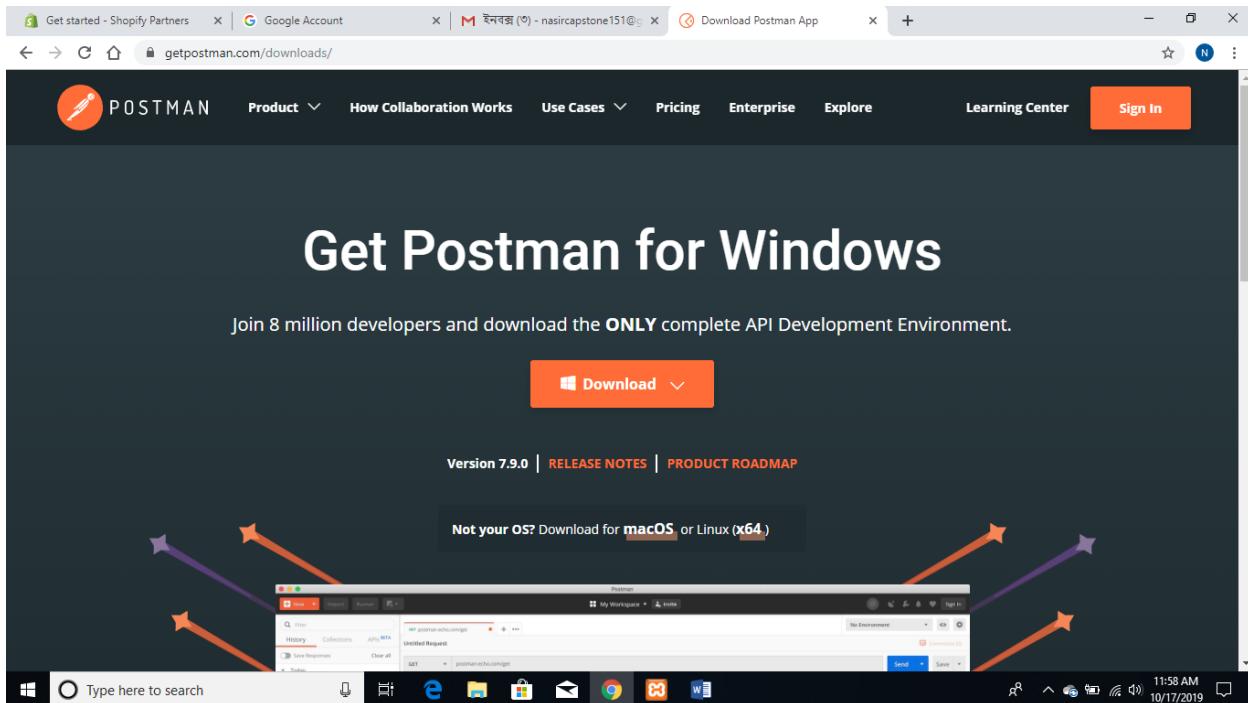
Now download **Windows installer(.msi) 32/64 bit** according to your computer's configuration.

Go to <https://ngrok.com/download>

The screenshot shows the ngrok download page. At the top, there are tabs for HOW IT WORKS, PRICING, DOWNLOAD, DOCS, and ENTERPRISE SOLUTIONS. The DOWNLOAD tab is highlighted. There is a LOGIN button and a SIGN UP button. The main section is titled "Download & setup ngrok" and says "Get started with ngrok in just a few seconds.". It is divided into four numbered steps: 1. Download ngrok, 2. Unzip to install, 3. Connect your account, and 4. Fire it up. Step 1 shows download links for Mac OS X, Linux, and Windows (32-bit). Step 2 shows a terminal command: `unzip /path/to/ngrok.zip`. Step 3 shows a terminal command: `./ngrok authtoken <YOUR_AUTH_TOKEN>` and a link to "Sign up for free". Step 4 shows a terminal command: `./ngrok http 80`. The bottom of the page shows a Windows taskbar with the date and time as 10/17/2019 at 11:55 AM.

Download according to your operating system. Here I have used windows operating, so I pressed **Download for Windows**.

Go to <https://www.getpostman.com/downloads/>



Press **Download** to get postman.

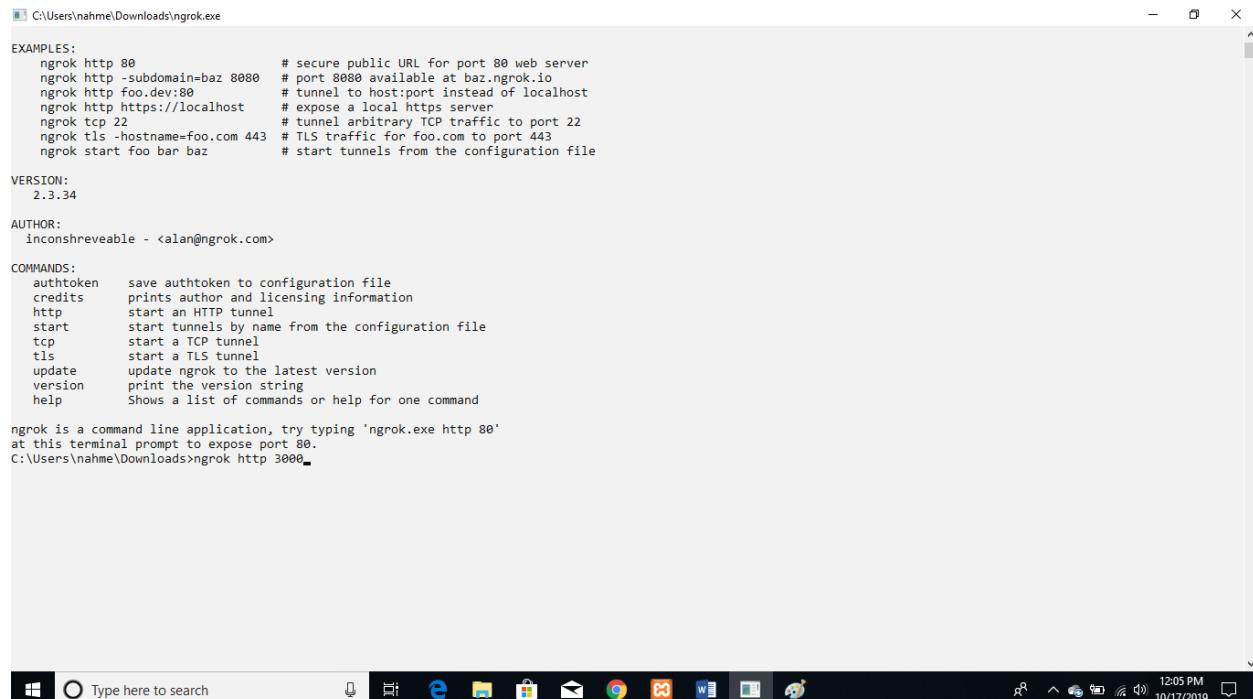
### **Task Number 3: Build a Shopify app with node and express**

Follow this tutorial link provided by Shopify <https://help.shopify.com/en/api/tutorials/build-a-shopify-app-with-node-and-express>

#### Step 1=> Expose your local development environment to the internet

Open ngrok.exe

Put=> **ngrok http 3000**



```
C:\Users\nahme\Downloads\ngrok.exe

EXAMPLES:
  ngrok http 80          # secure public URL for port 80 web server
  ngrok http -subdomain=foo 8080
  ngrok http foo.dev:80
  ngrok http https://localhost
  ngrok tcp 22
  ngrok tls -hostname=foo.com 443
  ngrok start foo bar baz      # start tunnels from the configuration file

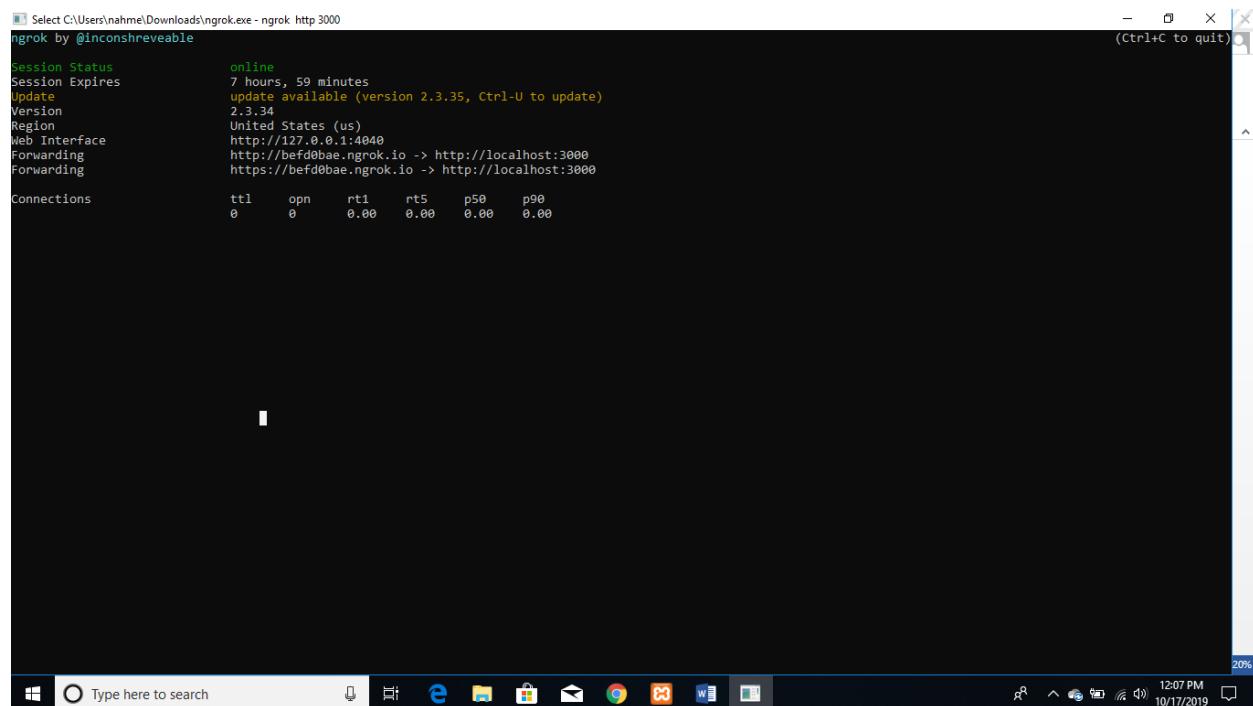
VERSION:
  2.3.34

AUTHOR:
  inconshreveable - <alan@ngrok.com>

COMMANDS:
  authtoken    save authtoken to configuration file
  credits      prints author and licensing information
  http         start an HTTP tunnel
  start        start tunnels by name from the configuration file
  tcp          start a TCP tunnel
  tls          start a TLS tunnel
  update       update ngrok to the latest version
  version      print the version string
  help         Shows a list of commands or help for one command

ngrok is a command line application, try typing 'ngrok.exe http 80'
at this terminal prompt to expose port 80.
C:\Users\nahme\Downloads>ngrok http 3000
```

Press Enter



```
Select C:\Users\nahme\Downloads\ngrok.exe - ngrok http 3000
ngrok by @inconshreveable

session Status          online
Session Expires        7 hours, 59 minutes
Update                 update available (version 2.3.35, Ctrl-U to update)
Version                2.3.34
Region                 United States (us)
Web Interface          http://127.0.0.1:4040
Forwarding             http://befd0bae.ngrok.io -> http://localhost:3000
Forwarding             https://befd0bae.ngrok.io -> http://localhost:3000

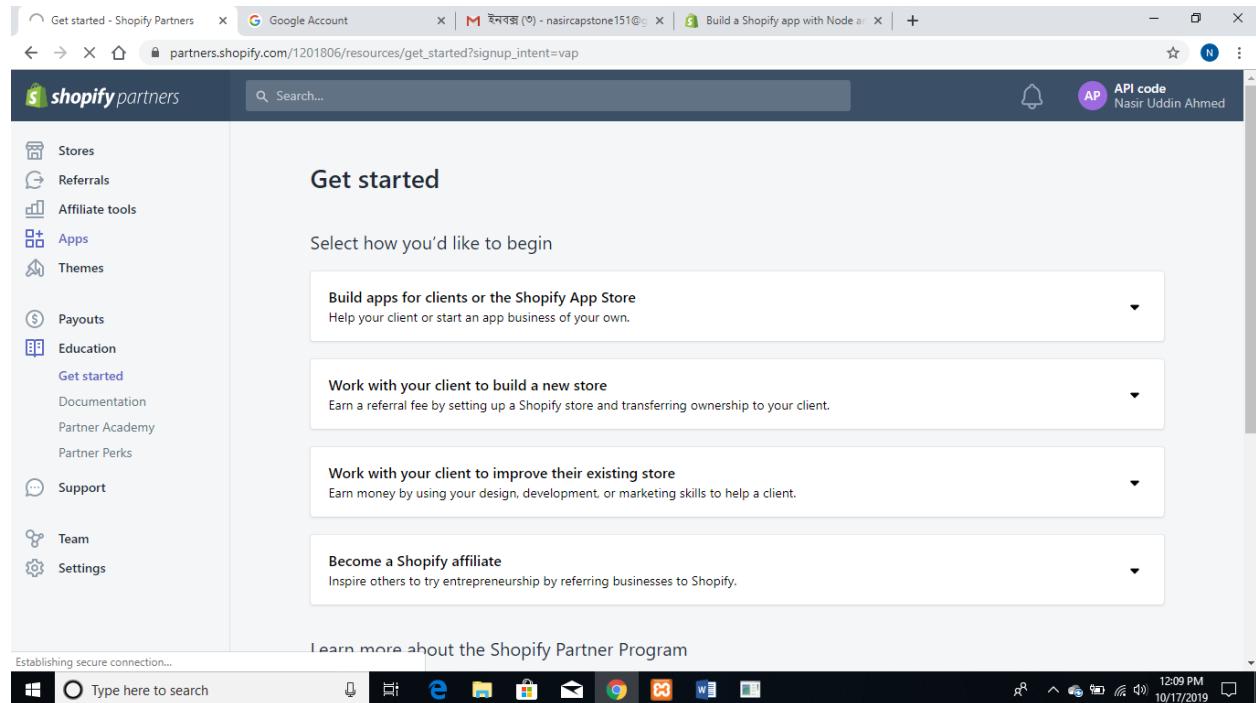
Connections            ttl     opn      rti      rt5      p50      p90
                        0       0       0.00    0.00    0.00    0.00

20%
```

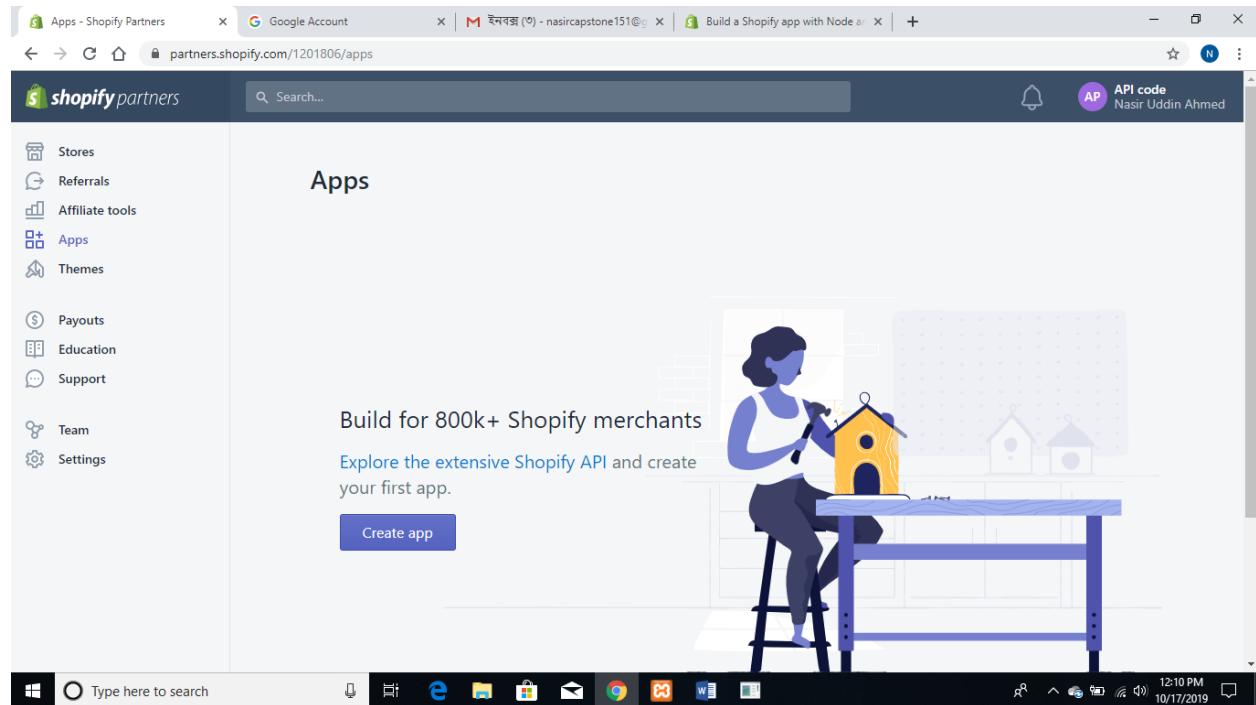
ngrok has generated tunneling address.

## Step 2=> Create and configure your app in the Partner Dashboard

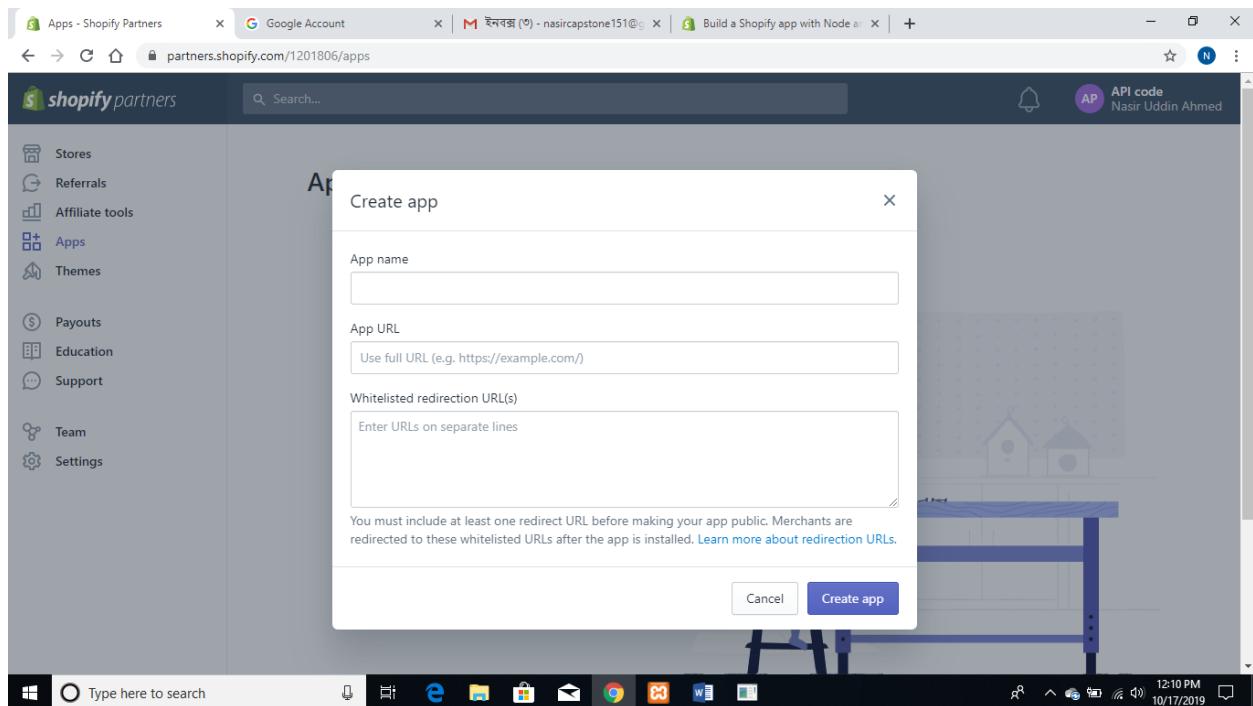
Go to your Shopify partner account. Press Apps



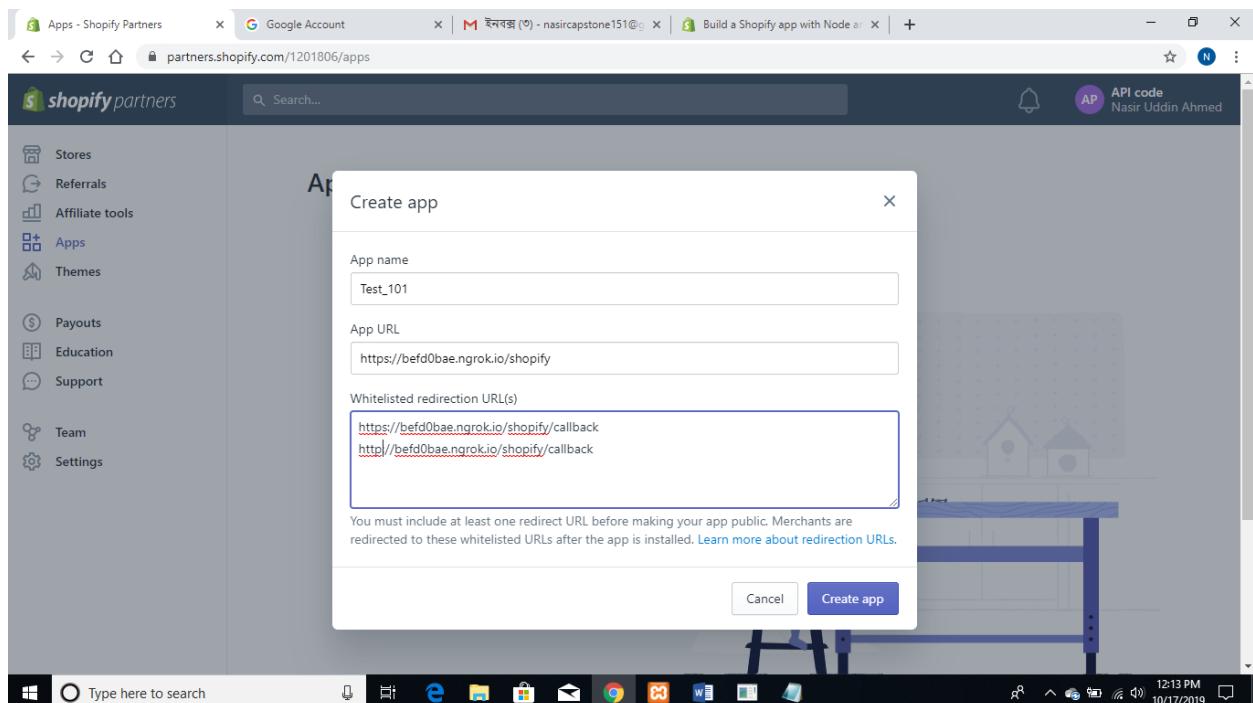
After pressing apps, a new window will appear.



Now press Create app



Here we need to put a App name, App URL generated by ngrok (Forwarding address). And Whitelisted redirection URL.

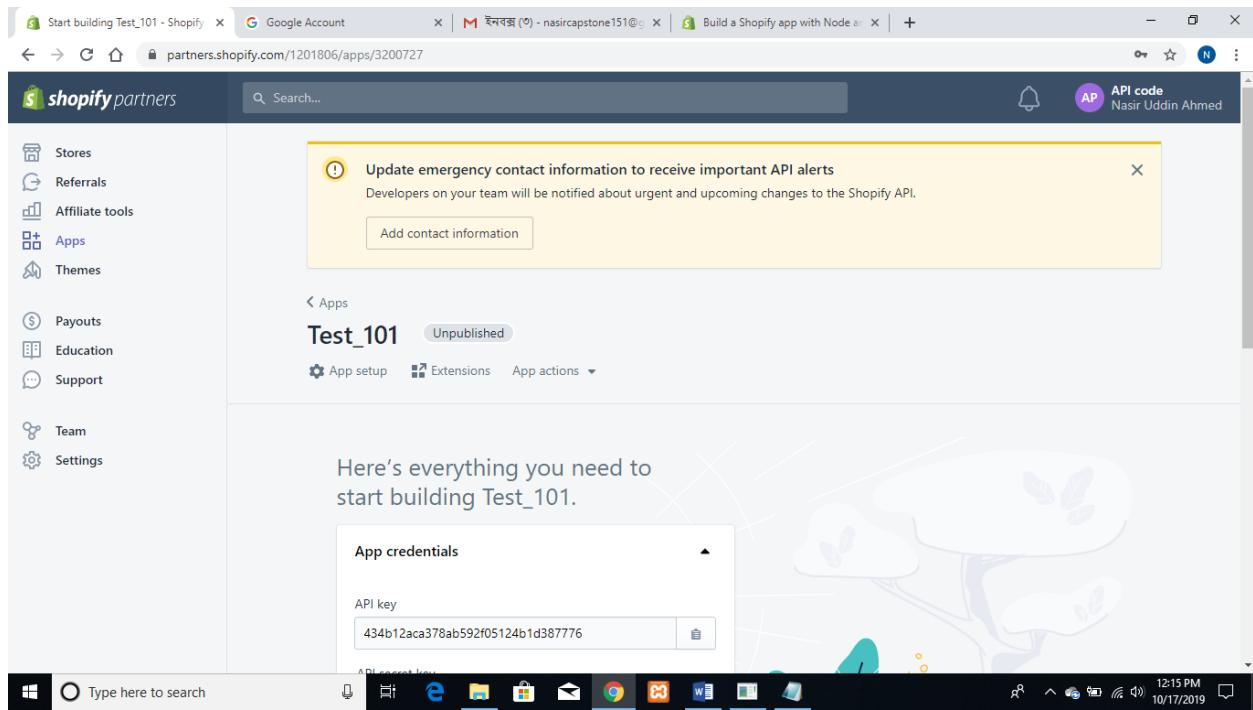


I have put my information according to my ngrok.

In App URL after .io put /shopify

In whitelisted URL after .io put /shopify/callback

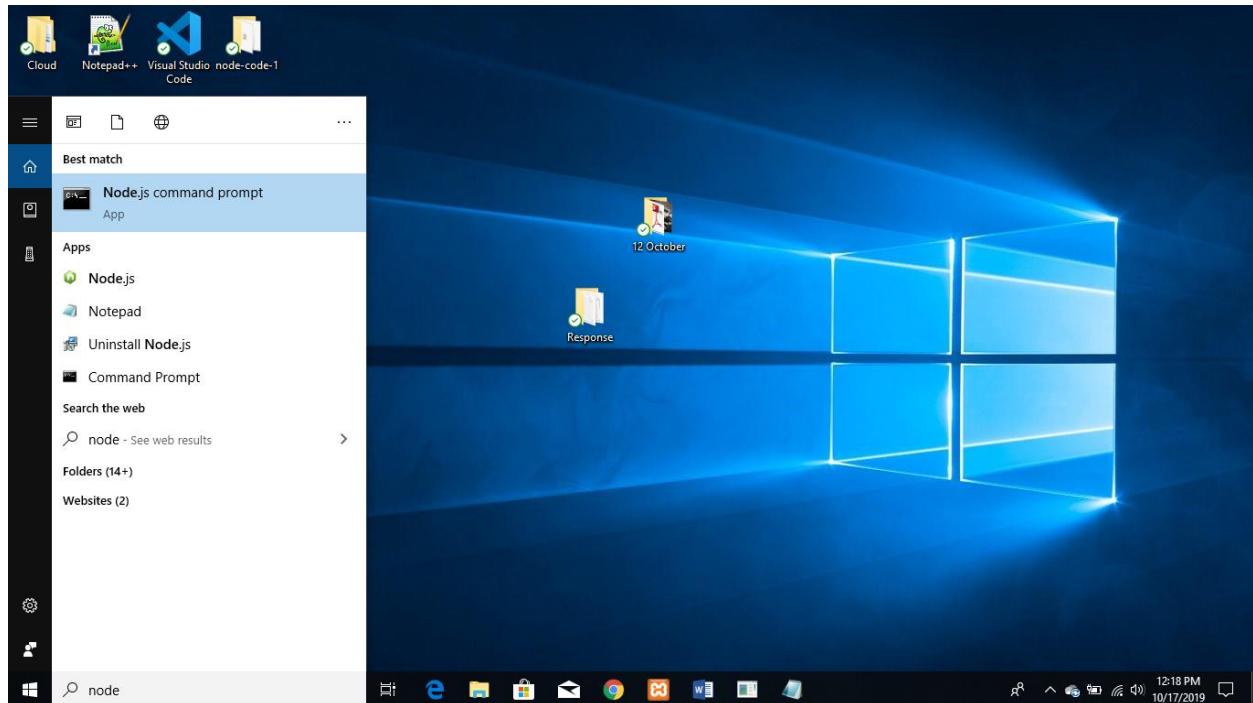
Now press Create app



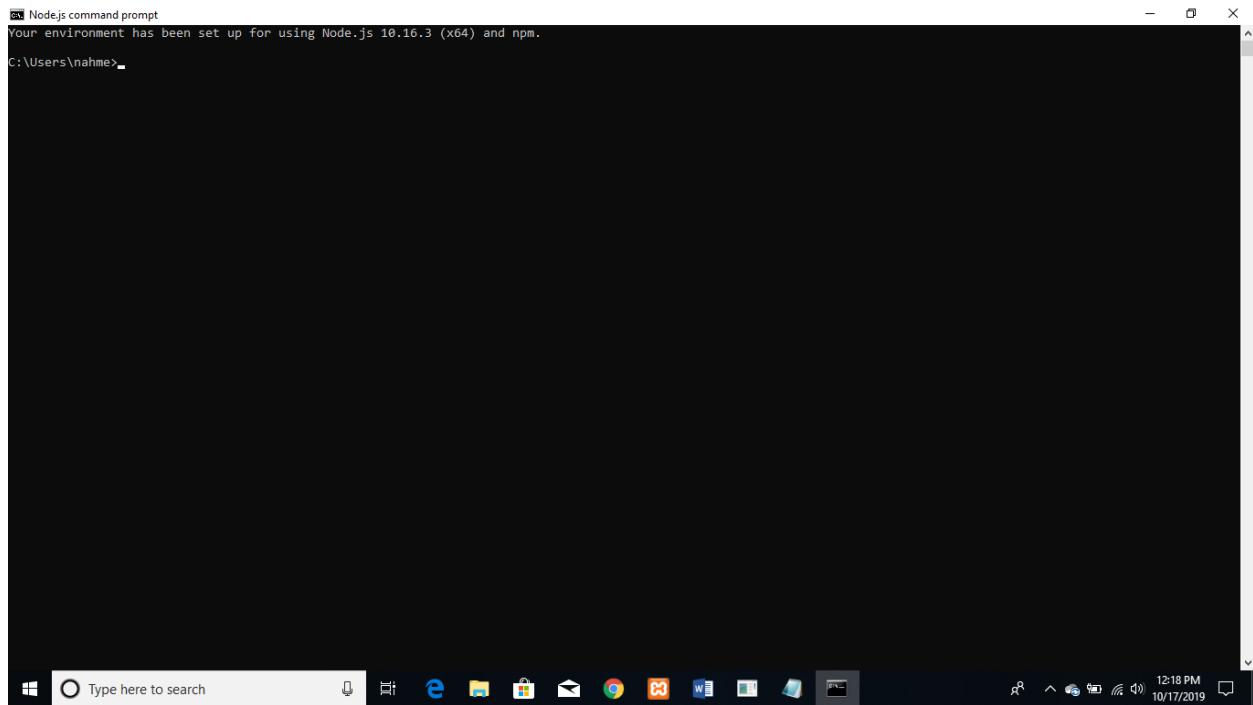
Here you will get your application's API Key, Secret key

### Step 3=> Create a Node.js Project

Open Node.js command prompt



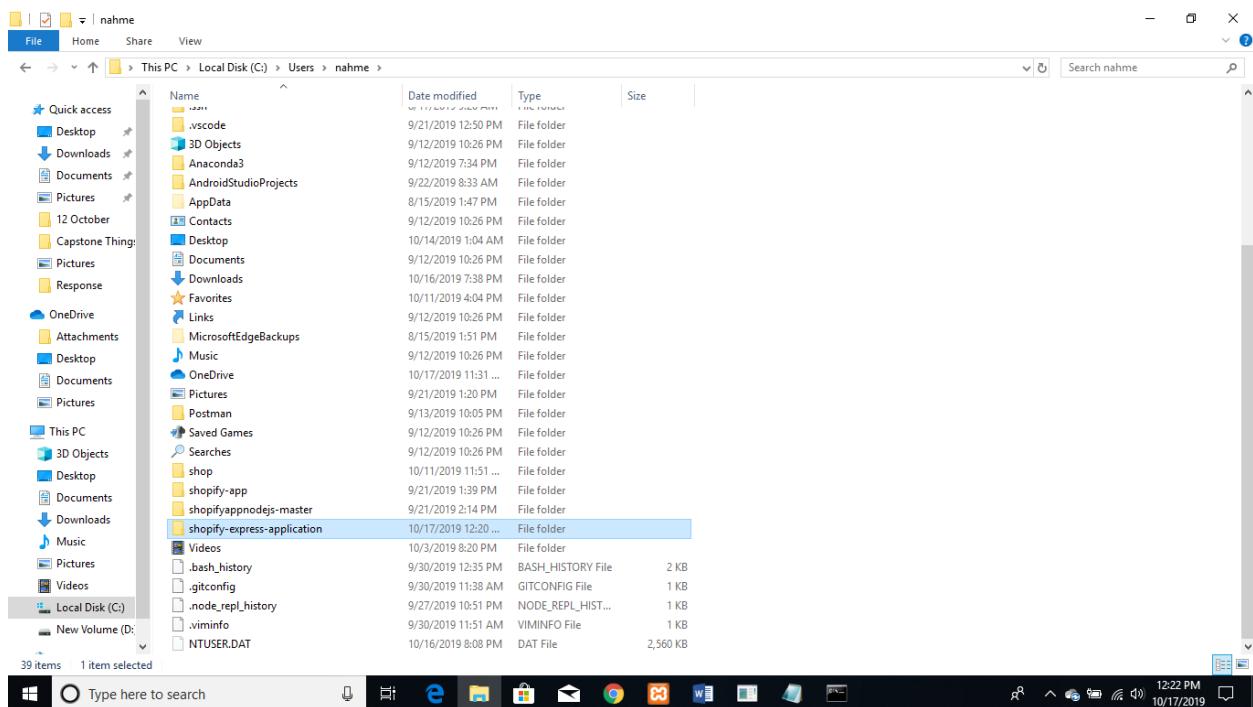
After clicking Node.js command prompt, the following window will come.



Now write 3 commands sequentially

Number 1: `mkdir shopify-express-application`

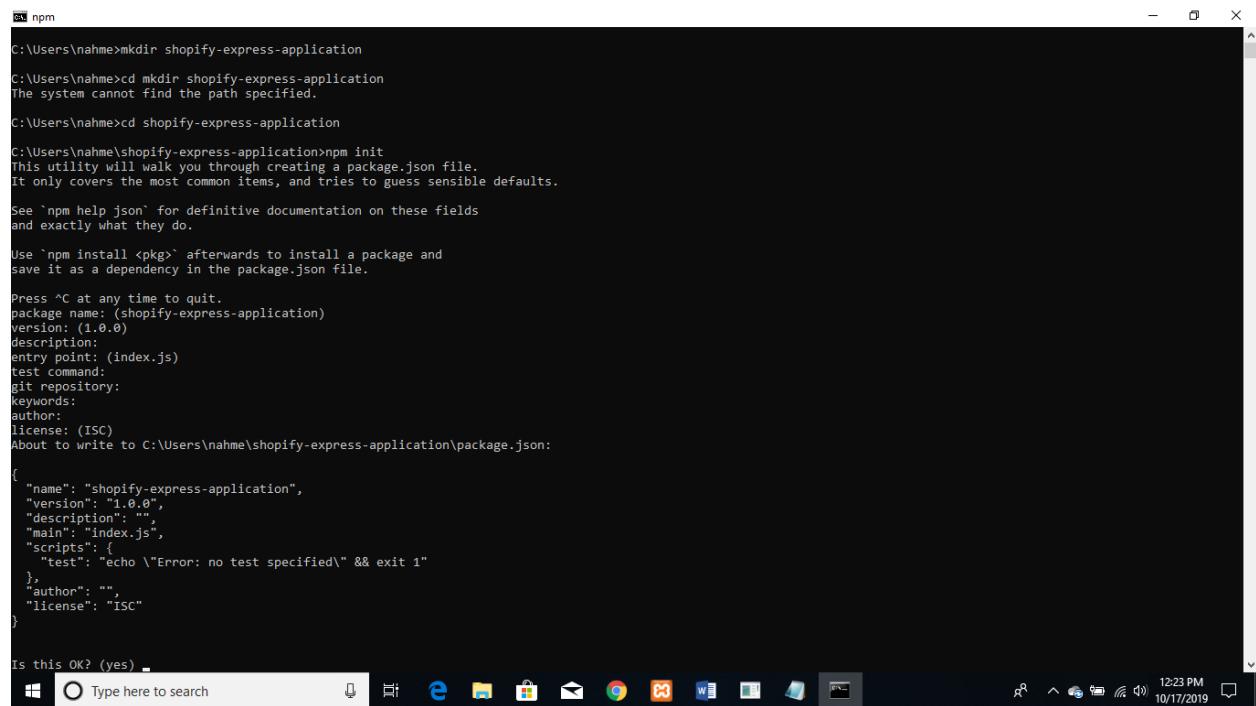
Number 2: `cd shopify-express-application`



Number 1 & Number 2 will create a Node.js project folder

Number 3: npm init

After giving npm init ,



```
npm
C:\Users\nahme>mkdir shopify-express-application
C:\Users\nahme>cd shopify-express-application
The system cannot find the path specified.

C:\Users\nahme>cd shopify-express-application
C:\Users\nahme\shopify-express-application>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
See 'npm help json' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

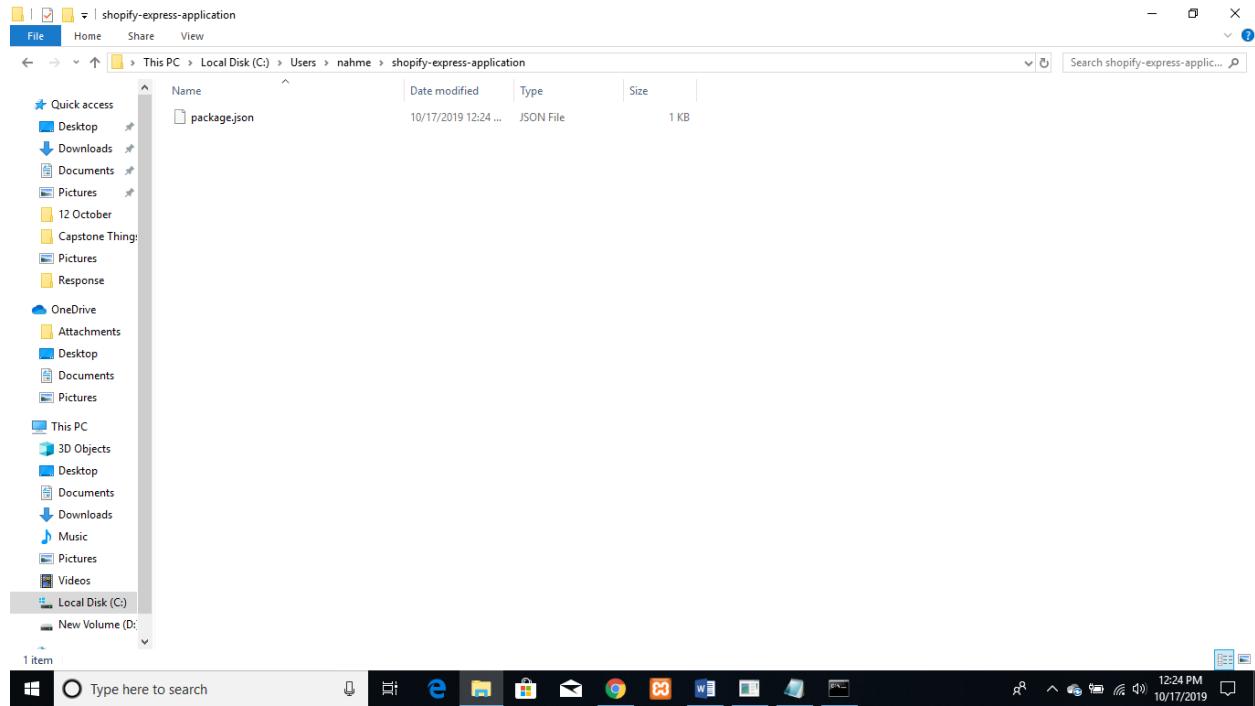
Press ^C at any time to quit.
package name: (shopify-express-application)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to C:\Users\nahme\shopify-express-application\package.json:

{
  "name": "shopify-express-application",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this OK? (yes) -
```

This will happen, input yes. And press enter.

Then a package.json file would be generated inside the Node.js project folder.



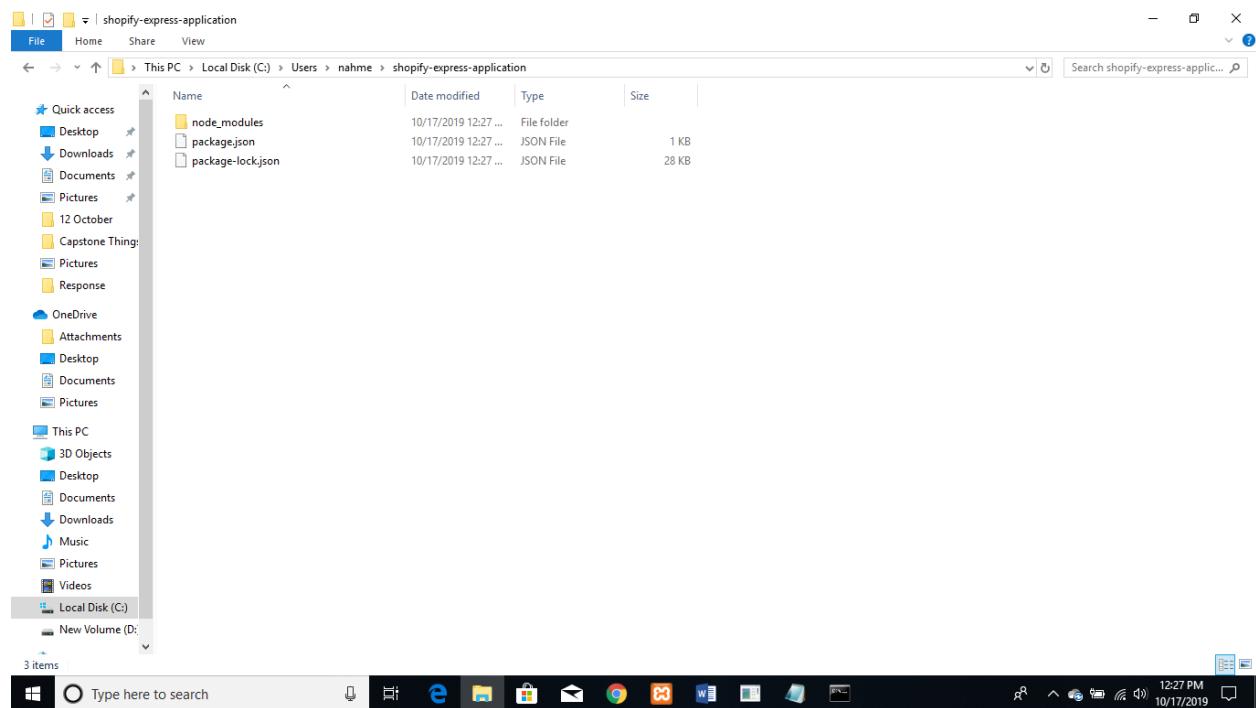
Documentation for npm init available in : <https://docs.npmjs.com/cli/init>

Now we need to install npm

Put `npm install express dotenv cookie nonce request request-promise --save`

A screenshot of a Windows Command Prompt window titled 'npm'. The command entered is 'npm install express dotenv cookie nonce request request-promise --save'. The output shows the progress of the installation, with '[.....]' indicating the completion of the process. The taskbar at the bottom is identical to the one in the previous screenshot.

This command will install necessary node modules and package-lock.json,



Put `npm install dotenv`

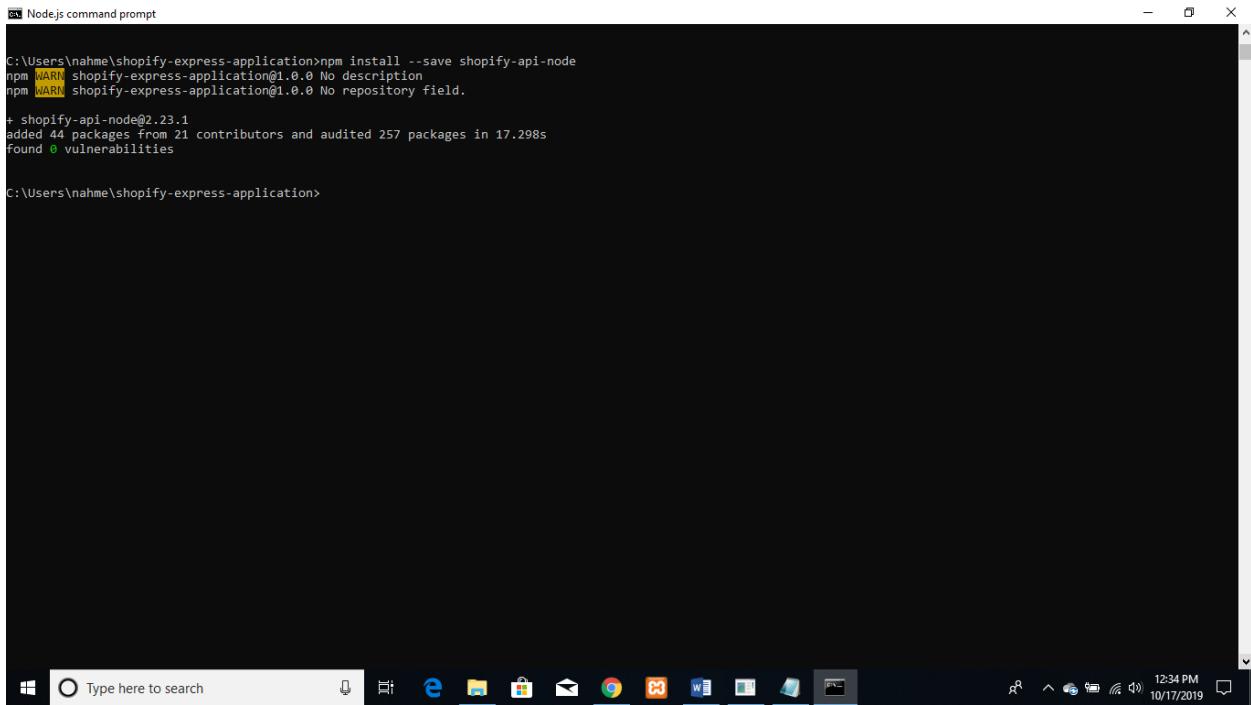
Documentation for dotenv is available : <https://www.npmjs.com/package/dotenv>

```
C:\Users\nahme\shopify-express-application>npm install dotenv
[.....] | loadIdealTree:loadAllDepsIntoIdealTree: sill install loadIdealTree
```

A screenshot of a Windows Command Prompt window titled "npm". The command "npm install dotenv" is being run in the directory "C:\Users\nahme\shopify-express-application". The output shows the command being run and the process of loading the ideal tree for all dependencies. The taskbar at the bottom includes icons for File Explorer, Edge, File Explorer, Mail, Google Chrome, and others.

Put `npm install --save shopify-api-node`

Documentation for shopify-node-api available: <https://www.npmjs.com/package/shopify-api-node>



```
Node.js command prompt
C:\Users\nahme\shopify-express-application>npm install --save shopify-api-node
npm WARN shopify-express-application@1.0.0 No description
npm WARN shopify-express-application@1.0.0 No repository field.

+ shopify-api-node@2.23.1
added 44 packages from 21 contributors and audited 257 packages in 17.298s
found 0 vulnerabilities

C:\Users\nahme\shopify-express-application>
```

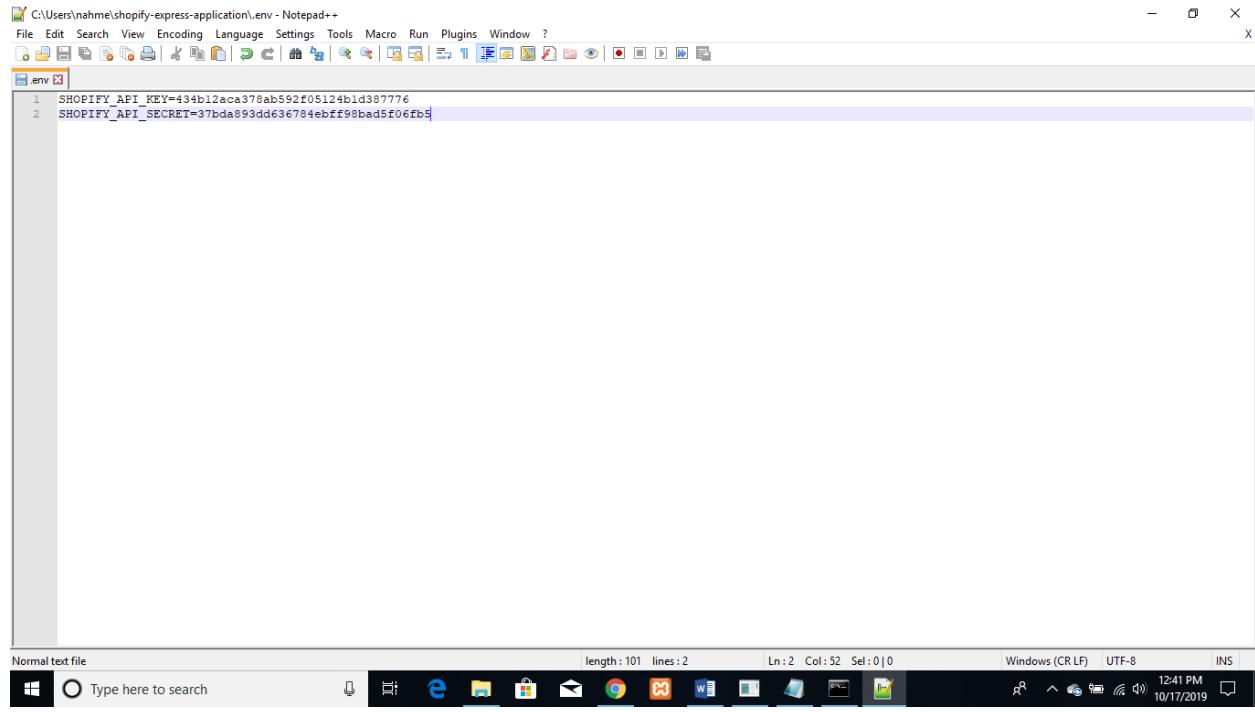
Now we need to create a .env file

Put `touch .env` (Sometimes touch would not work, it will show => touch is not recognized as an internal or external command, <https://stackoverflow.com/questions/36126269/touch-is-not-recognized-as-an-internal-or-external-command-operable-program-o>), in that case use `npm install touch-cli -g`

In the .env file, put

```
SHOPIFY_API_KEY="{YOUR_API_KEY}"
SHOPIFY_API_SECRET="{YOUR_API_SECRET_KEY}"
```

API KEY and API SECRET KEY available on Shopify partner account dashboard.



C:\Users\nahme\shopify-express-application\.env - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

env [3]

```
1 SHOPIFY_API_KEY=434b12aca378ab592f05124b1d387776
2 SHOPIFY_API_SECRET=37bda893dd636784ebff98bad5f06fb$
```

Normal text file length : 101 lines : 2 Ln:2 Col:52 Sel:0|0 Windows (CR LF) UTF-8 INS

Type here to search 12:41 PM 10/17/2019

Now create a .gitignore file

Put `touch .gitignore`

In the .gitignore file type .env

#### Step 4=> Start building your Node.js app

Now create a index.js file

Put `touch index.js`

In the index.js file , copy the following code,

```
const dotenv = require('dotenv').config();
const express = require('express');
const app = express();
const crypto = require('crypto');
const cookie = require('cookie');
const nonce = require('nonce')();
const querystring = require('querystring');
const request = require('request-promise');
```

```

const apiKey = process.env.SHOPIFY_API_KEY;
const apiSecret = process.env.SHOPIFY_API_SECRET;
const scopes = 'read_products';
const forwardingAddress = "{ngrok forwarding address}"; // Replace this with
your HTTPS Forwarding address

app.get('/', (req, res) => {
  res.send('Hello World!');
});

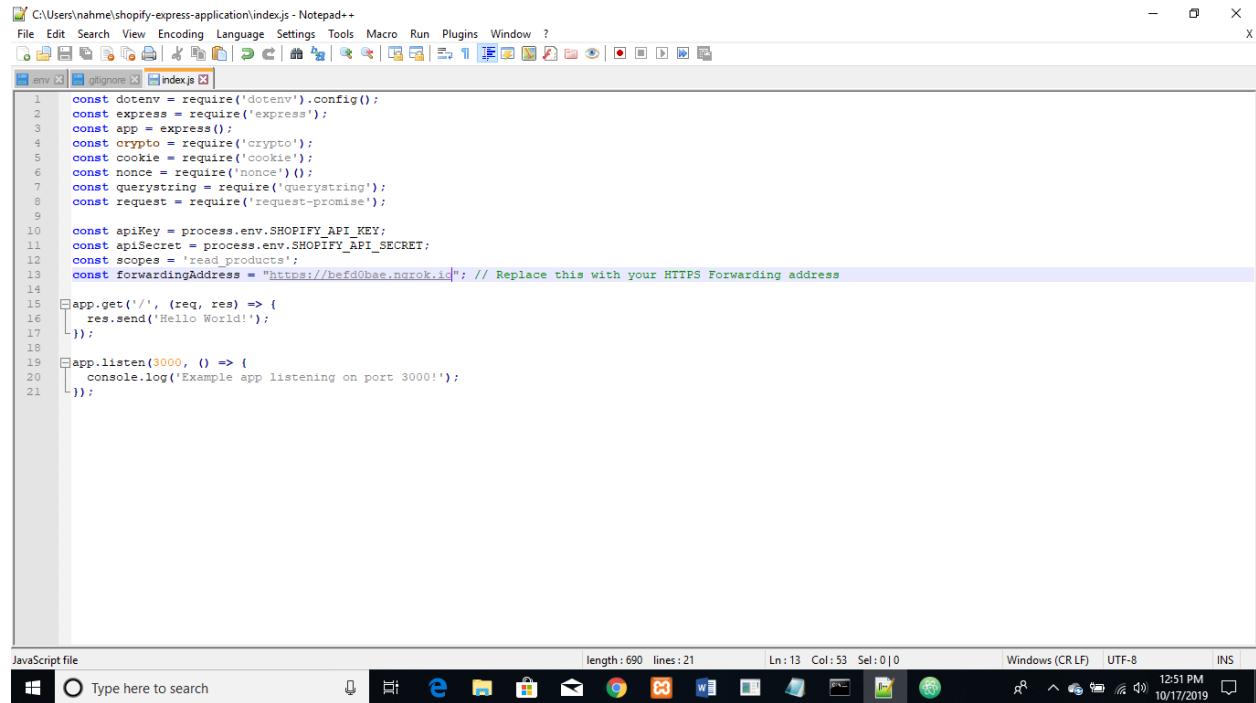
app.listen(3000, () => {
  console.log('Example app listening on port 3000!');
});

```

Here=>

**const** forwardingAddress = "{ngrok forwarding address}"; replace the  
forwarding address, put the address that has been generated by ngrok.

Your index.js code file will look like



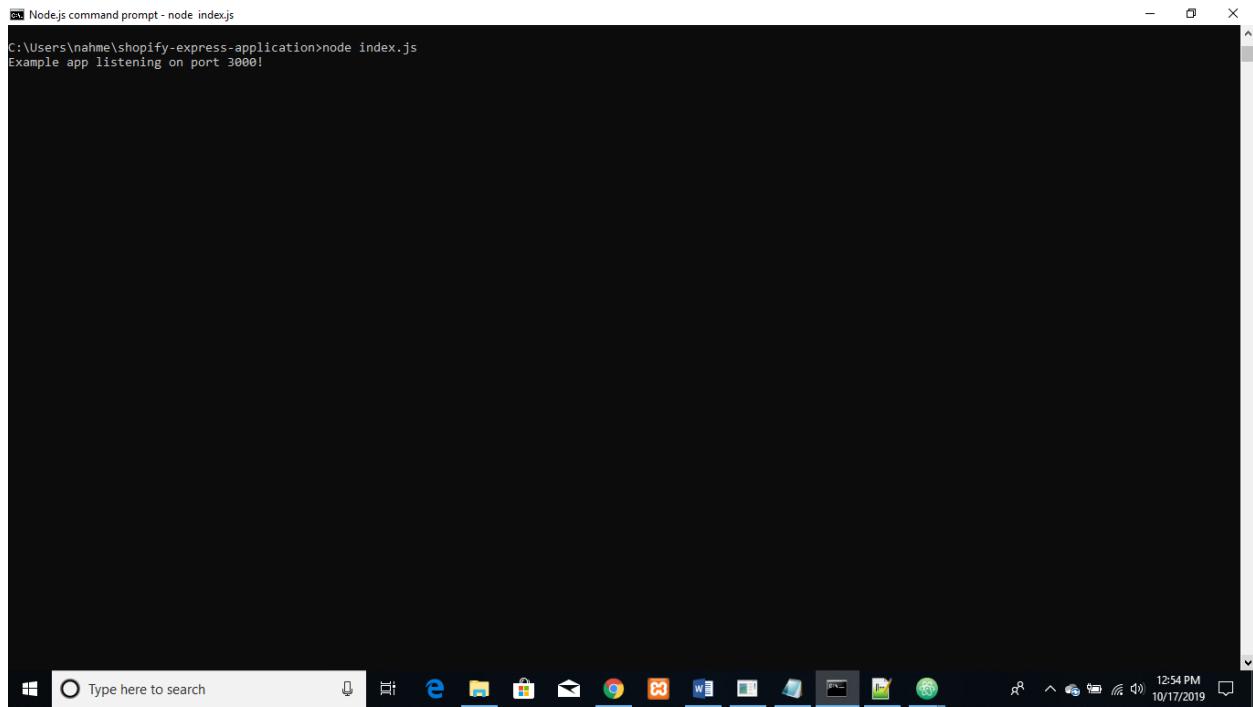
```

1  const dotenv = require('dotenv').config();
2  const express = require('express');
3  const app = express();
4  const crypto = require('crypto');
5  const cookie = require('cookie');
6  const nonce = require('nonce')();
7  const querystring = require('querystring');
8  const request = require('request-promise');
9
10 const apiKey = process.env.SHOPIFY_API_KEY;
11 const apiSecret = process.env.SHOPIFY_API_SECRET;
12 const scopes = 'read_products';
13 const forwardingAddress = "https://befd0bae.ngrok.io"; // Replace this with your HTTPS Forwarding address
14
15 app.get('/', (req, res) => {
16   res.send('Hello World!');
17 });
18
19 app.listen(3000, () => {
20   console.log('Example app listening on port 3000!');
21 });

```

Now you need to run the Node.js application to see the tunneling.

Put `node index.js`

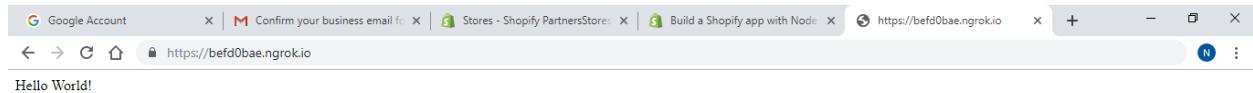


```
Node.js command prompt - node index.js
C:\Users\nahme\shopify-express-application>node index.js
Example app listening on port 3000!
```

`Node index.js` would run the application.

Go to your browser and type the forwarding address generated by ngrok, (<https://befd0bae.ngrok.io>)

It will show a Hello World String.

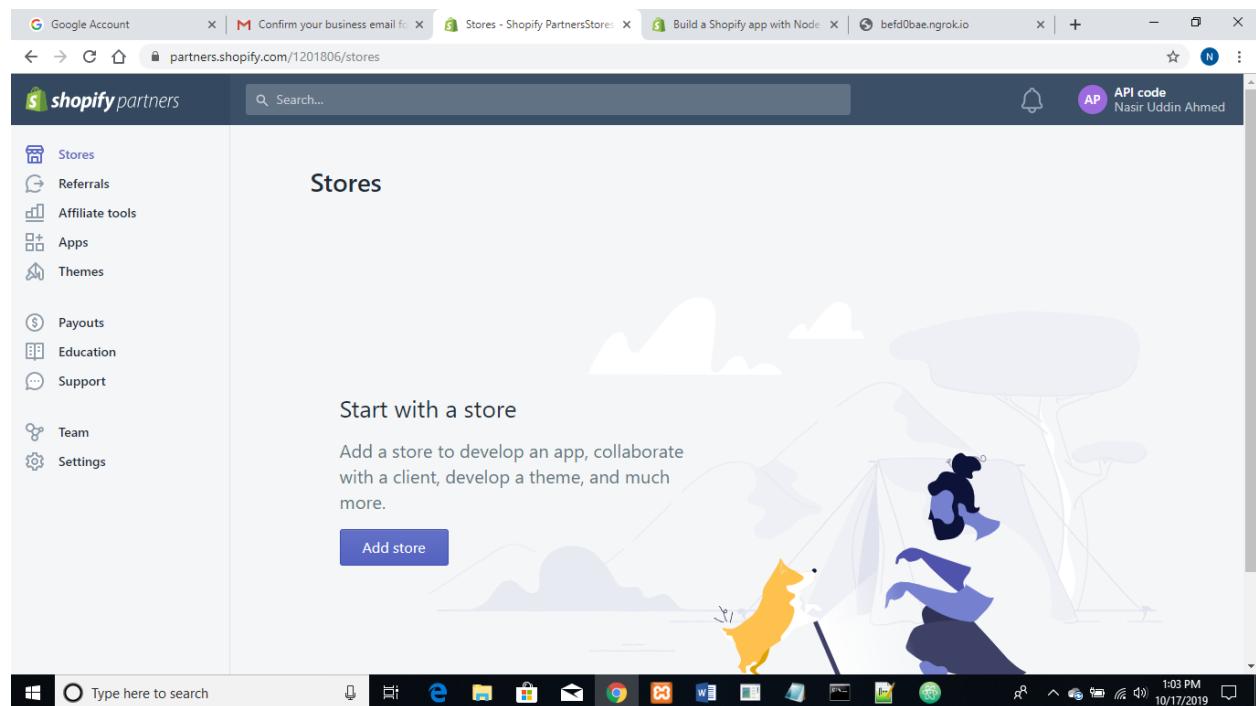


If the code fails, then

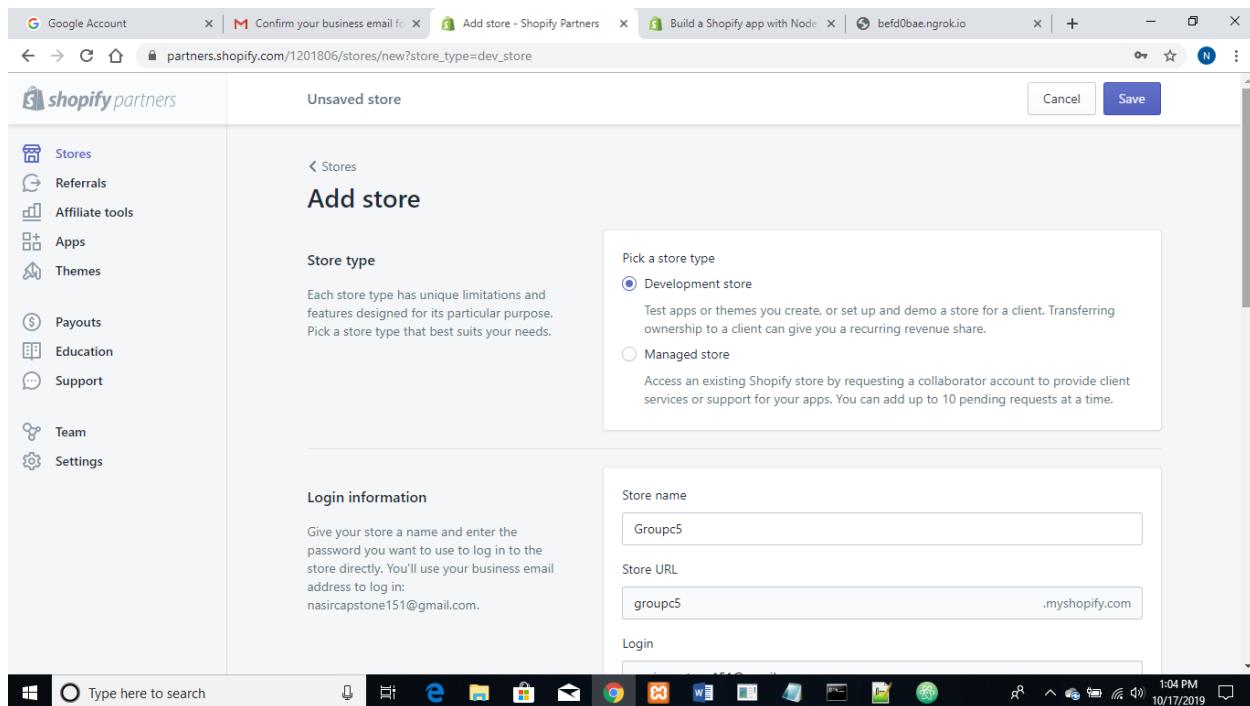
```
const scopes = 'read_products'; replace this line with const scopes =
'write_products';
in the index.js file, then save and run the project by using node index.js
command from Node.js command prompt
```

### Step 5=> Create Shopify-specific routes

Before creating specific route we need to set a Shopify store. Go to Shopify partner dashboard, click on **Add store**.

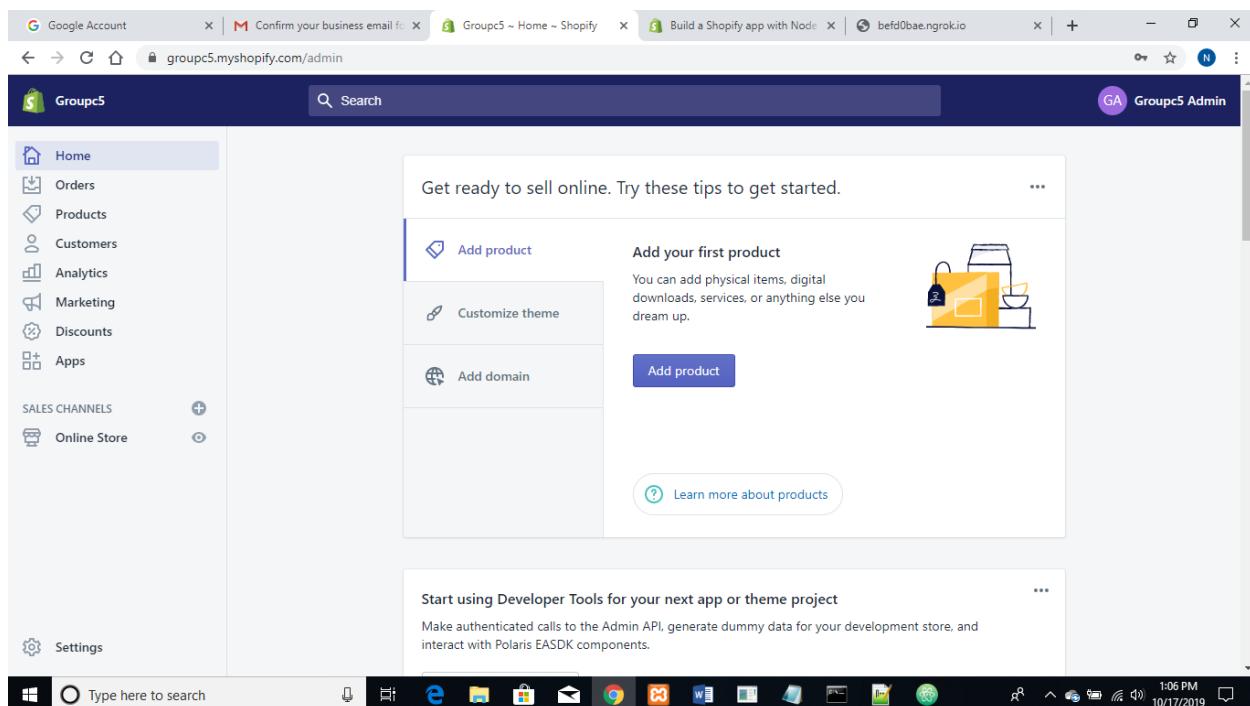


Fill up necessary information, shown below. Here I gave a store name, you can give yours.



Then press **save**

You will be redirected to your store's admin panel.



Now go to index.js file, replace the previous code with the given code below

```
const dotenv = require('dotenv').config();

const express = require('express');

const app = express();

const crypto = require('crypto');

const cookie = require('cookie');

const nonce = require('nonce')();

const querystring = require('querystring');

const request = require('request-promise');

const apiKey = process.env.SHOPIFY_API_KEY;

const apiSecret = process.env.SHOPIFY_API_SECRET;

const scopes = 'write_products'; // both privileg

const forwardingAddress = "https://5dec032c.ngrok.io"; // Replace this with your HTTPS Forwarding address

/*app.get('/', (req, res) => {

  res.send('Hello World!, I Am Nasir Uddin Ahmed From Capstone C5 Group, I Am A Software Engineer');

});*/



app.get('/shopify', (req, res) => {

  const shop = req.query.shop;

  if (shop) {

    const state = nonce(); // constant noce

    const redirectUri = forwardingAddress + '/shopify/callback';

    const installUrl = 'https://' + shop +

      '/admin/oauth/authorize?client_id=' + apiKey +

      '&scope=' + scopes +

      '&state=' + state +

      '&redirect_uri=' + redirectUri; // hit back , source
```

```
res.cookie('state', state);

res.redirect(installUrl);

} else {

  return res.status(400).send('Missing shop parameter. Please add ?shop=your-development-
shop.myshopify.com to your request');

}

});

/*app.get('/shopify/callback', (req, res) => {

const { shop, hmac, code, state } = req.query;

const stateCookie = cookie.parse(req.headers.cookie).state;

if (state !== stateCookie) {

  return res.status(403).send('Request origin cannot be verified');

}

if (shop && hmac && code) {

  res.status(200).send('Callback route');

  // TODO

  // Validate request is from Shopify

  // Exchange temporary code for a permanent access token

  // Use access token to make API call to 'shop' endpoint

  // Custom coding started form here

  const map=Object.assign({}, req.query); // Custom Coding

  delete map['hmac']; // Custom Coding

  const message=querystring.stringify(map); // Custom Coding

}
```

```
const generatedHash=crypto
.createHmac('sha256',apiSecret) // my secret apiKey
.update(message)
.digest('hex');

if(generatedHash!==hmac){
    return res.status(400).send('HMAC Validaiton failed');
}

return res.status(200).send('HMAC Validated');

} else {
    res.status(400).send('Required parameters missing');
}
});*/
app.get('/shopify/callback', (req, res) => {
    const { shop, hmac, code, state } = req.query;
    const stateCookie = cookie.parse(req.headers.cookie).state;

    if (state !== stateCookie) {
        return res.status(403).send('Request origin cannot be verified');
    }

    if (shop && hmac && code) {
        // DONE: Validate request is from Shopify
        const map = Object.assign({}, req.query);
        delete map['signature'];
        delete map['hmac'];
    }
}
```

```
const message = querystring.stringify(map);

const providedHmac = Buffer.from(hmac, 'utf-8');

const generatedHash = Buffer.from(
  crypto
    .createHmac('sha256', apiSecret)
    .update(message)
    .digest('hex'),
  'utf-8'
);

let hashEquals = false;

try {
  hashEquals = crypto.timingSafeEqual(generatedHash, providedHmac)
} catch (e) {
  hashEquals = false;
}

if (!hashEquals) {
  return res.status(400).send('HMAC validation failed');
}

// DONE: Exchange temporary code for a permanent access token

const accessTokenRequestUrl = 'https://' + shop + '/admin/oauth/access_token';

const accessTokenPayload = {
  client_id: apiKey,
  client_secret: apiSecret,
  code,
};


```

```
request.post(accessTokenRequestUrl, { json: accessTokenPayload })

.then((accessTokenResponse) => {

  const accessToken = accessTokenResponse.access_token;
  // DONE: Use access token to make API call to 'shop' endpoint

  const shopRequestUrl = 'https://' + shop + '/admin/api/2019-04/products.json';
  //const shopRequestUrl = 'https://' + shop + '/admin/shop.json';

  const shopRequestHeaders = {

    'X-Shopify-Access-Token': accessToken, // Verified

  };

  request.get(shopRequestUrl, { headers: shopRequestHeaders })

  .then((shopResponse) => {

    res.status(200).end(shopResponse);

  })

  .catch((error) => {

    res.status(error.statusCode).send(error.error.error_description);

  });

  .catch((error) => {

    res.status(error.statusCode).send(error.error.error_description);

  });

}

} else {

  res.status(400).send('Required parameters missing');

}

});

app.listen(3000, () => {
```

```
console.log('Example app listening on port 3000!');

});
```

Careful , don't forget to change the forwarding address.

```
const forwardingAddress = "https://5dec032c.ngrok.io"; // Replace this with your HTTPS Forwarding address
```

Save index.js

And now run the index.js file again by using `node index.js` command.

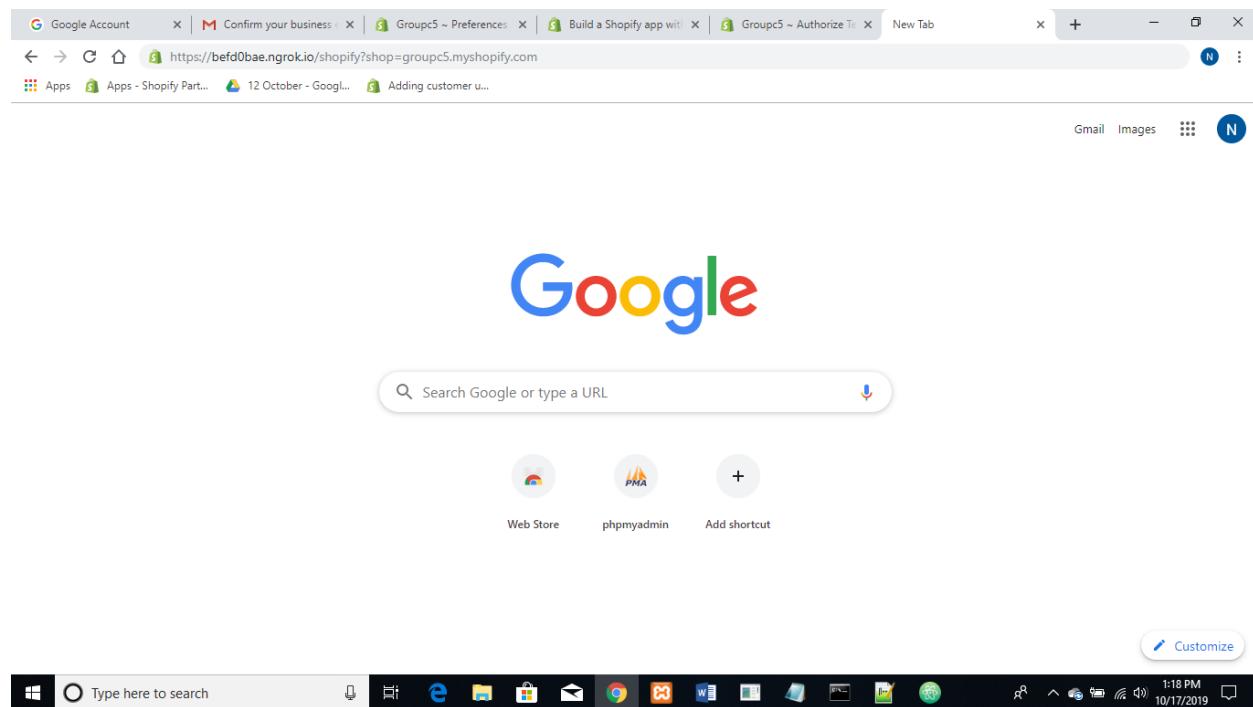
#### Step 6=> Run the application

Go to your browser

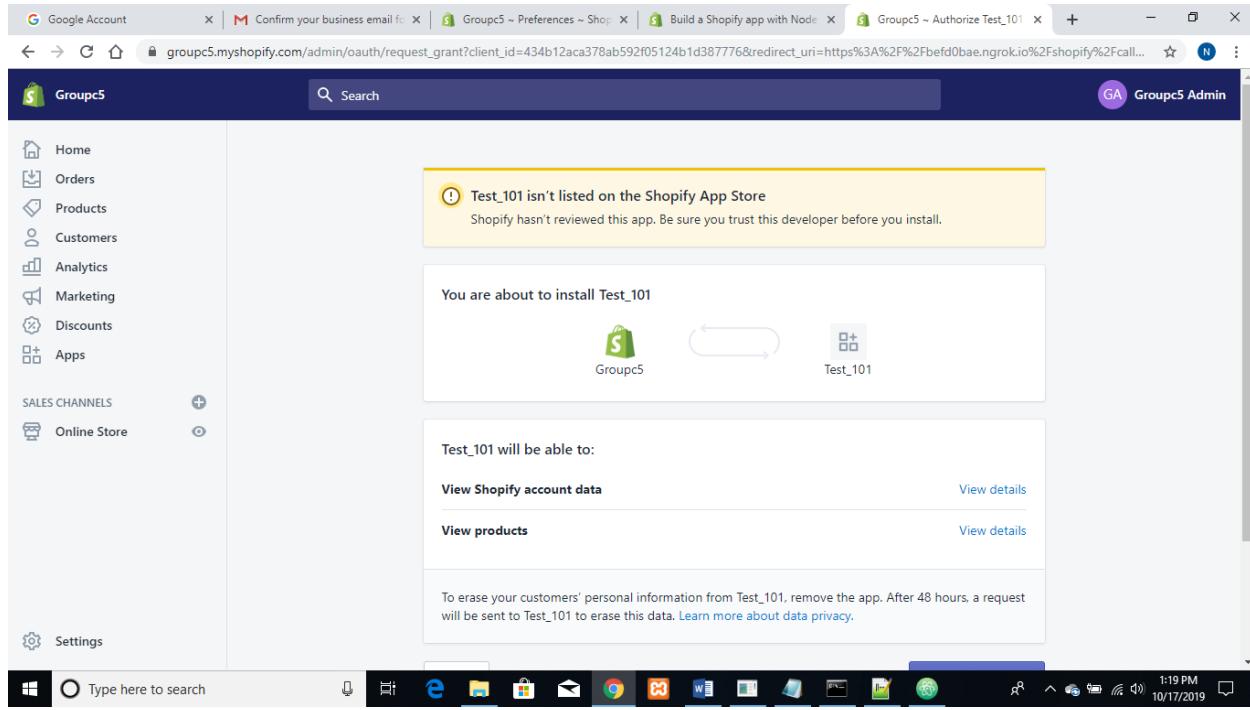
```
{your ngrok forwarding address}/shopify?shop=your-development-shop.myshopify.com
```

For our project it looked like

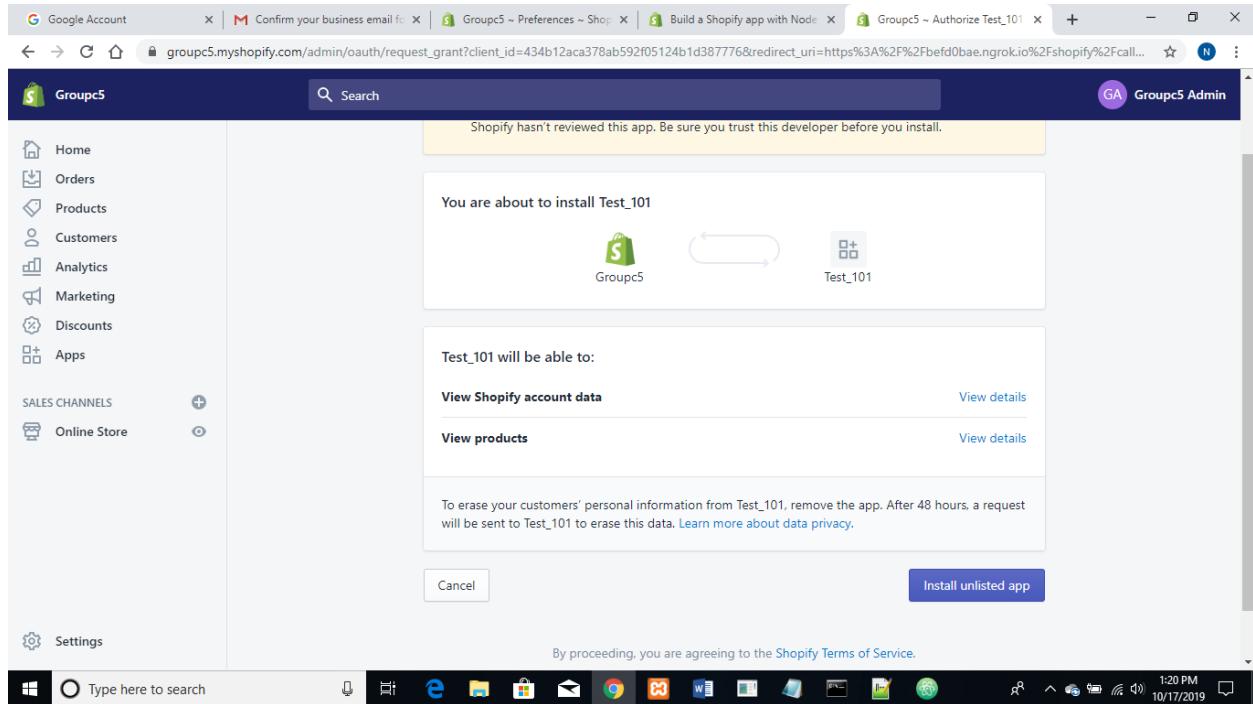
<https://befd0bae.ngrok.io/shopify?shop=groupc5.myshopify.com>



Put that link that has been showed in the above picture. Wait for some time. Tunneling forwarding needs time to detect.



If all ok, congratulations!!!, you have successfully created a bridge, forwarding for your Shopify application.



Press Install unlisted app



The page will show empty product list, as there is no product in the store.

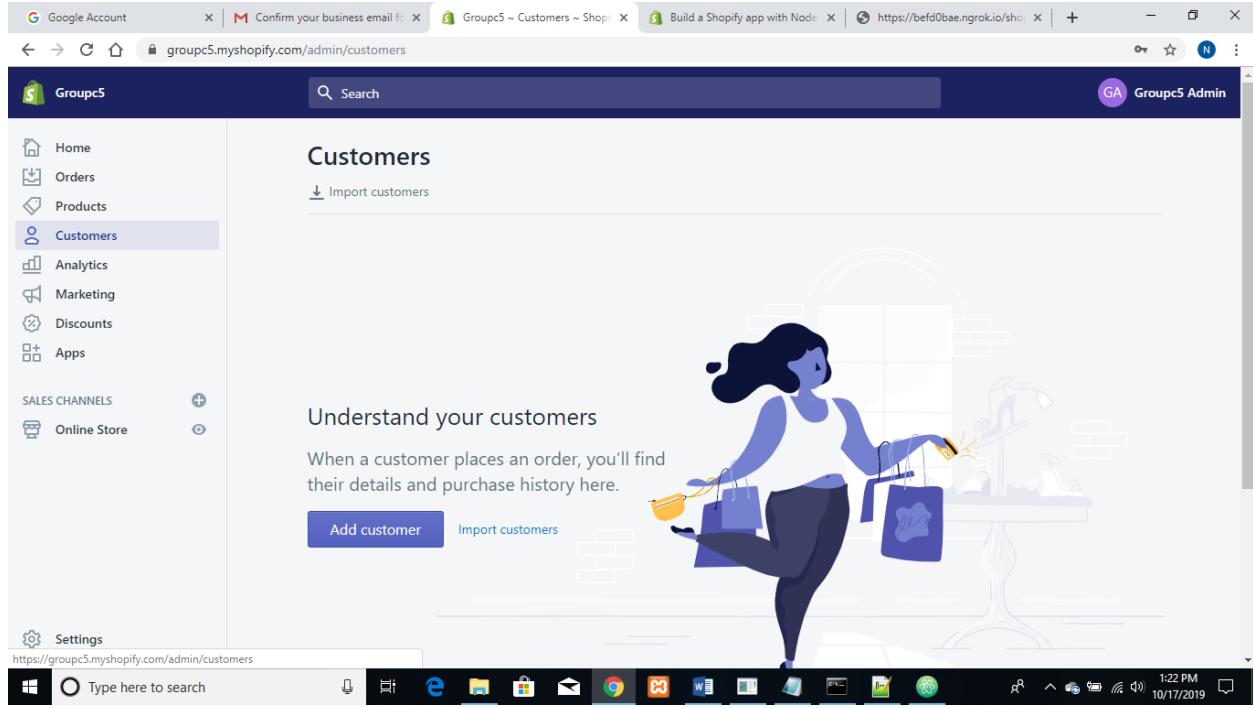
The screenshot shows the Shopify Admin interface for managing apps. The left sidebar includes links for Home, Orders, Products, Customers, Analytics, Marketing, Discounts, and Apps (which is selected). Under Sales Channels, there's an Online Store link. The main content area is titled 'Apps' and displays a card for 'Test\_101'. The card includes a thumbnail, the app name, an 'About' button, and a trash icon. Below this, a section encourages exploring more apps with a 'Visit the Shopify App Store' button. A note at the bottom says 'Working with a developer on your shop? Manage private apps' and a 'Learn more about apps' link.

By clicking Apps, you can see the application

The screenshot shows the details page for the 'Test\_101' app. The left sidebar is identical to the previous screenshot. The main content area shows the app card with 'Test\_101' and its code 'by API code'. The app's code is displayed as a JSON object: `{"products":[]}`. The Windows taskbar at the bottom shows various open applications like File Explorer, Edge, and Google Chrome.

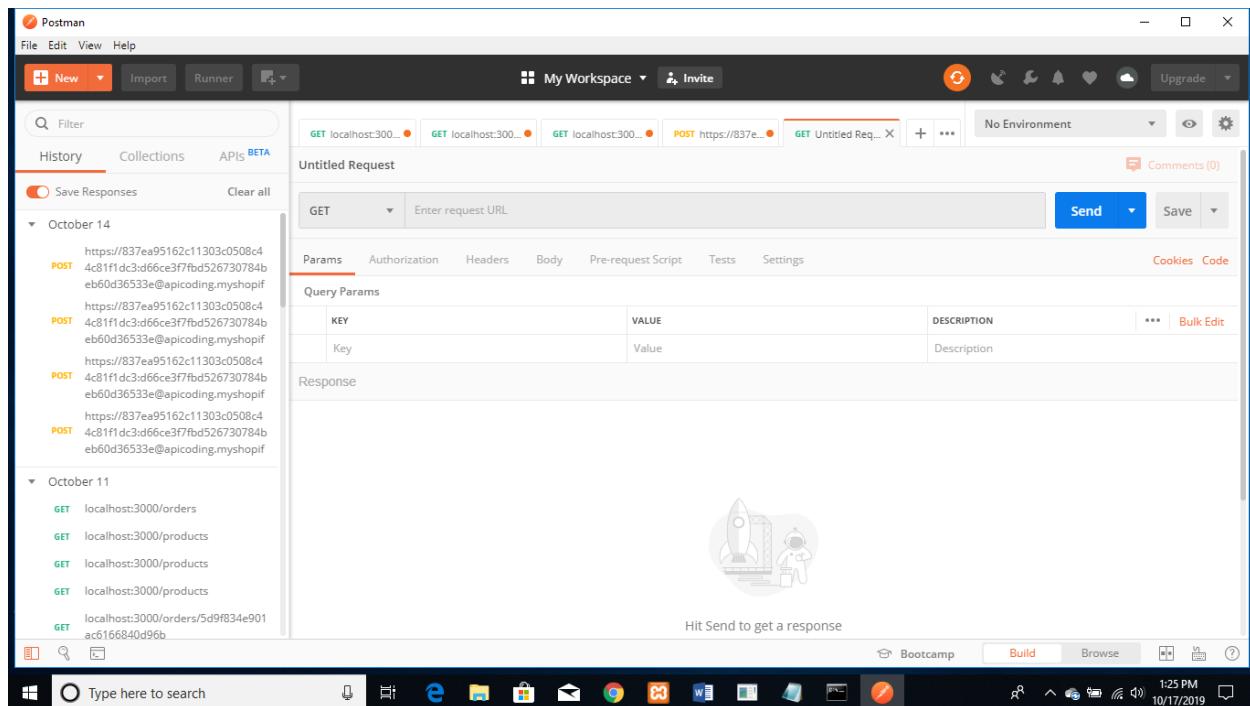
Now test the app, using User registration.

# User Registration By API Calling



By clicking customers, we can see there is no customer at present. To call the API, we need postman.

Open postman app.



In the postman app, we have to select POST request, (because we want to register a user)

Go to <https://accounts.shopify.com/store-login>, here you will get example URL, that is consisted with access token, and credentials.

The screenshot shows the Shopify Admin dashboard with the 'Apps' section selected in the sidebar. The main content area displays a card for an app named 'Test\_101' with options to 'About' or delete it. Below this, there's a section to 'Explore 2,500+ more apps' with a link to 'Visit the Shopify App Store'. At the bottom, there's a note for developers about managing private apps and a link to learn more about apps. The bottom taskbar includes the Windows Start button, a search bar, and various system icons.

Click on manage private apps.

The screenshot shows the Shopify Admin interface with the following details:

- Header:** Google Account, Confirm your business, Groupc5 ~ Account, Groupc5 ~ Private apps, https://befd0bae.ngrok.io, Example URL - Shopify.
- Sidebar:** Home, Orders, Products, Customers, Analytics, Marketing, Discounts, Apps (selected).
- Top Bar:** Search, Private apps, Groupc5 Admin.
- Section:** Create private app
- App details:** Private app name (input field), Emergency developer email (input field), Learn more.
- Admin API:** Your API credentials will be generated when you Save.
  - Store content like articles, blogs, comments, pages, and redirects (read\_content, write\_content): Read access
  - Customer details and customer groups (read\_customers, write\_customers): Read access
  - Orders, transactions and fulfillments (read\_orders, write\_orders): Read access
- Bottom:** Type here to search, taskbar with various icons, 1:31 PM, 10/17/2019.

Give the App name that you used before, In our case it was Test\_101

Make the read access to Read and Write Access.

Put a tick mark on Allow this app to access your storefront data using the Storefront API. Press save.

Click the Apps option, there you will see your app. Click app(Test\_101).

The screenshot shows the Shopify Admin interface with the 'Private apps' section open. The left sidebar includes links for Home, Orders, Products, Customers, Analytics, Marketing, Discounts, and Apps (which is selected). Under Sales Channels, 'Online Store' is listed. The main content area displays a table with one row for a private app named 'Test\_101'. The table columns are 'Private app name', 'API key', and 'Contact email'. The API key is '9ec6d2641ec2fdf1a38c953a5871d731' and the contact email is 'nahmed151086@bscse.uiu.ac.bd'. A blue button at the top right says 'Create a new private app'. Below the table, a note states 'Private apps are subject to the Shopify API License and Terms of Use.' and a link to 'Learn more about Private apps.'

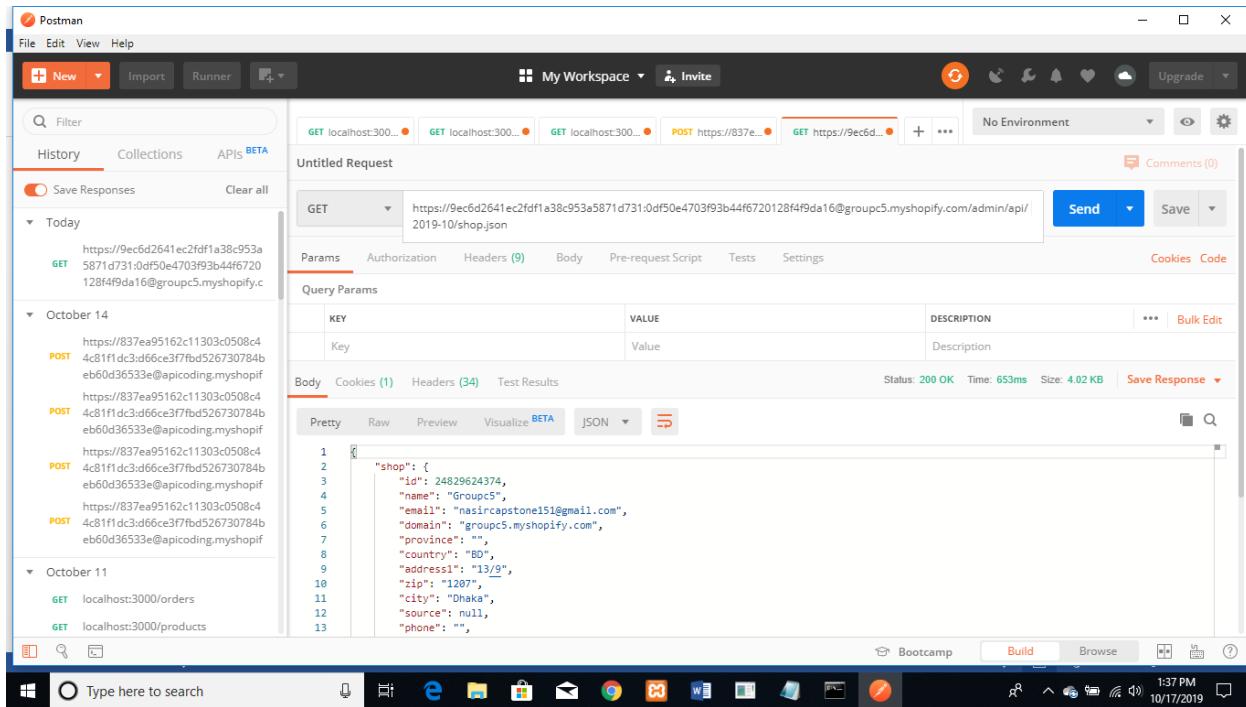
Clicking on Test\_101 will show you a new page

The screenshot shows the configuration page for the private app 'Test\_101'. The left sidebar is identical to the previous screenshot. The main content area has two tabs: 'Admin API' and 'ADMIN API PERMISSIONS'. The 'Admin API' tab is active, showing fields for 'API key' (set to '9ec6d2641ec2fdf1a38c953a5871d731'), 'Password' (a masked password), and 'Example URL' (set to 'https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-04')). Below these fields, a note says 'Private applications authenticate with Shopify through basic HTTP authentication, using the URL format https://[apikey]:[password]@[hostname]/admin/api/[version]/[resource].json'. The 'Shared Secret' field contains '64d7bb0faea9984f45e5fd8688b85230'. The 'ADMIN API PERMISSIONS' tab is shown below, with a note about storing content like articles, blogs, comments, pages, and redirects.

From here, Example URL is important. Copy that.

Go to Postman and put the example URL.

<https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-10/shop.json>.

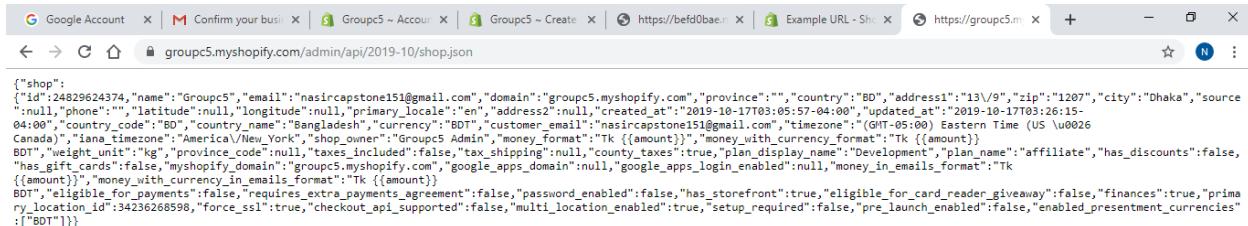


The screenshot shows the Postman application interface. In the center, there is an 'Untitled Request' section for a GET request to the URL: https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-10/shop.json. The 'Params' tab is selected. Below the request details, the response body is shown as a JSON object:

```
1 "shop": {  
2     "id": "24829624374",  
3     "name": "Groupc5",  
4     "email": "nasircapstone151@gmail.com",  
5     "domain": "groupc5.myshopify.com",  
6     "province": "",  
7     "country": "BD",  
8     "address1": "13/9",  
9     "zip": "1207",  
10    "city": "Dhaka",  
11    "source": null,  
12    "phone": ""},  
13 }
```

This will show the api calling.

Go to your browser and put the example URL, you can GET the shop information.



```
{"shop": {"id": 24829624374, "name": "Groupcs", "email": "nasircapstone151@gmail.com", "domain": "groupcs.myshopify.com", "province": "", "country": "BD", "address1": "13/9", "zip": "1207", "city": "Dhaka", "source": "null", "phone": "", "latitude": null, "longitude": null, "primary_locale": "en", "address2": null, "created_at": "2019-10-17T03:05:57-04:00", "updated_at": "2019-10-17T03:26:15-04:00", "country_code": "BD", "country_name": "Bangladesh", "currency": "BDT", "customer_email": "nasircapstone151@gmail.com", "timezone": "(+05:00) Eastern Time (US \u0026 Canada)", "iana_timezone": "America/New_York", "shop_owner": "Groupcs Admin", "money_format": "Tk {{amount}}", "money_with_currency_format": "Tk {{amount}} BDT", "weight_unit": "kg", "province_code": null, "taxes_included": false, "tax_shipping": null, "county_taxes": true, "plan_display_name": "Development", "plan_name": "affiliate", "has_discounts": false, "has_gift_cards": false, "myshopify_domain": "groupcs.myshopify.com", "google_apps_domain": null, "google_apps_login_enabled": null, "money_in_emails_format": "Tk {{amount}} BDT", "eligible_for_payments": false, "requires_extra_payments_agreement": false, "password_enabled": false, "has_storefront": true, "eligible_for_card_reader_giveaway": false, "finances": true, "primary_location_id": 34236268598, "force_ssl": true, "checkout_api_supported": false, "multi_location_enabled": true, "setup_required": false, "pre_launch_enabled": false, "enabled_presentation_currencies": ["BDT"]}}
```



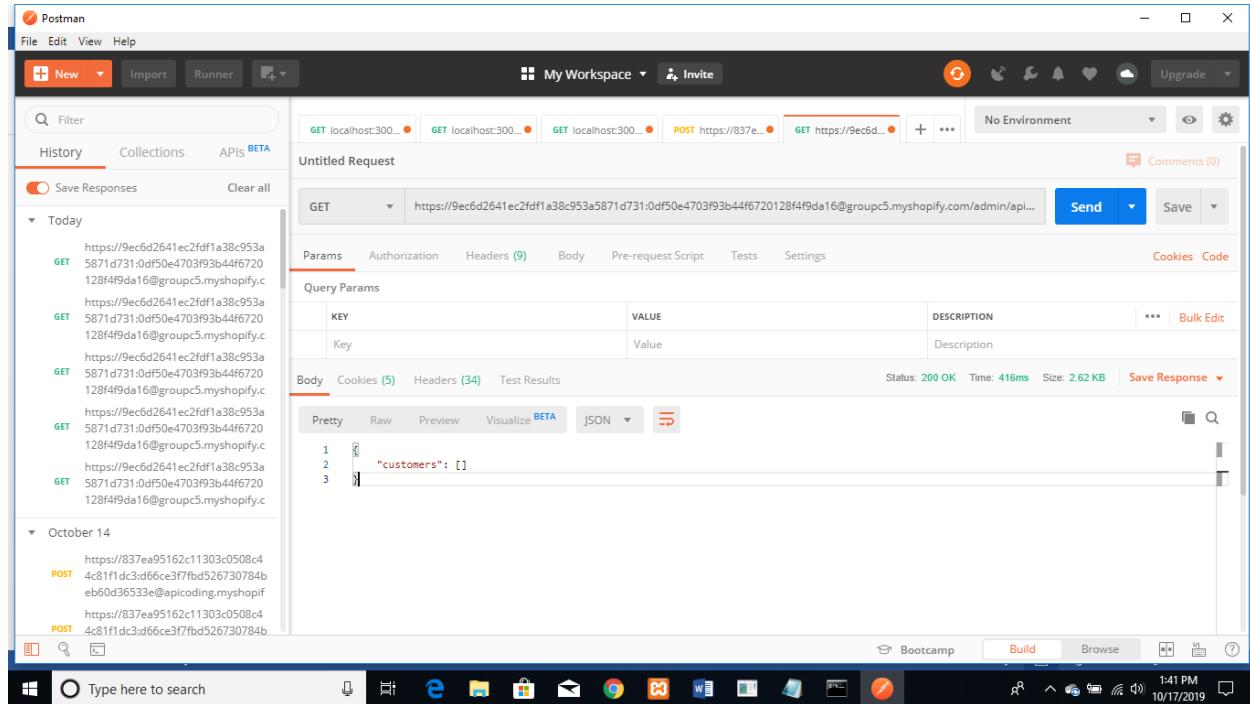
For registering an user.

Use the example URL in postman.

<https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-10/customers.json>

Note /api/2019-10/ after the slash, just put what API you need.

Initially there is no customer.



## Put

<https://9ec6d2641ec2fdf1a38c953a5871d731:0df50e4703f93b44f6720128f4f9da16@groupc5.myshopify.com/admin/api/2019-10/customers.json>

On postman, select Post.

Now copy the following request body in the postman.

```
{
  "customer": {
    "first_name": "Nasir Uddin",
    "last_name": "Ahmed",
    "email": "pmj011151086@gmail.com",
    "phone": "+15142546011",
    "verified_email": true,
    "addresses": [
      {
        "address1": "Mohammadpur",
        "city": "Dhaka",
        "state": "Dhaka"
      }
    ]
  }
}
```

```

    "province": "ON",
    "phone": "028110362",
    "zip": "1207",
    "last_name": "ABC",
    "first_name": "DEF",
    "country": "Bangladesh"
  },
],
"send_email_invite": true
}
}

```

This code is available on <https://community.shopify.com/c/Shopify-APIs-SDKs/Adding-customer-using-postman/td-p/345019>

The screenshot shows the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', 'Help', 'New', 'Import', 'Runner', and a search bar. The main workspace is titled 'Untitled Request' and shows a POST method pointing to a Shopify API endpoint: `https://9ec6d2641ec2fdf1a38c953a5871d7310df50e4703f93b44f6720128f49f9da16@groupc5.myshopify.c...`. The 'Body' tab is selected, displaying a JSON payload:

```

1 {  
2   "customer": {  
3     "first_name": "Nasir Uddin",  
4     "last_name": "Ahmed",  
5     "email": "pm301115108@gmail.com",  
6     "phone": "+1810009888",  
7     "verified_email": true,  
8     "addresses": [  
9       {  
10         "address1": "Mohammadpur",  
11         "city": "Dhaka",  
12         "province": "ON",  
13         "phone": "+028110362",  
14         "zip": "1207",  
15         "last_name": "ABC",  
16         "first_name": "DEF",  
17         "country": "Bangladesh"  
18       }  
19     ],  
20     "send_email_invite": true  
21   }  
22 }

```

The 'Params', 'Authorization', 'Headers', 'Tests', and 'Settings' tabs are also visible. Below the request details, there's a 'Response' section and a toolbar with 'Send', 'Save', and other options. The bottom of the screen shows the Windows taskbar with various pinned icons.

Then press **send**

Customer would be added to the Shopify store by api calling.

The screenshot shows the Shopify Admin interface. The left sidebar has 'Customers' selected. The main area is titled 'Customers' and shows a table with one row. The row contains the name 'ABC DEF BD', 0 orders, and Tk 0.00 spent. A message at the bottom says 'Customer accounts are disabled. Edit settings.'

Using GET request and customers API, we can get registered customer.

The screenshot shows Postman with a GET request to the Shopify Customers API. The response is a JSON object:

```

1  "customers": [
2    {
3      "id": 2575836610614,
4      "email": "turjoahmed75@gmail.com",
5      "accepts_marketing": false,
6      "created_at": "2019-10-17T03:48:52-04:00",
7      "updated_at": "2019-10-17T03:48:52-04:00",
8      "first_name": "ABC",
9      "last_name": "DEF",
10     "orders_count": 0,
11     "state": "disabled",
12     "total_spent": "0.00",
13     "last_order_id": null,
14     "note": "",
15     "verified_email": true,
16     "multipass_identifier": null,
17     "tax_exempt": false,
18     "phone": "+8801197129456",
19     "tags": "",
20     "last_order_name": null,
21     "currency": "BDT",
22     "addresses": [
23       {
24         "id": 2738799280182,
25         ...
26       }
27     ]
28   }
29 ]
30 
```

Code link of the project => <https://github.com/Turjo7/Integration-of-Multiple-Services-using-Application-Programming-Interface-API>

Thank You